

OFPPT

ROYAUME DU MAROC

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle et de la Promotion du Travail

DIRECTION RECHERCHE ET INGENIERIE DE FORMATION

RESUME THEORIQUE

&

GUIDE DE TRAVAUX PRATIQUES

MODULE N°17 : INITIATION AUX AUTOMATES PROGRAMMABLES

SECTEUR : FROID ET GENIE THERMIQUE

SPECIALITE : MAINTENANCE HOTELIERE

NIVEAU : TECHNICIEN

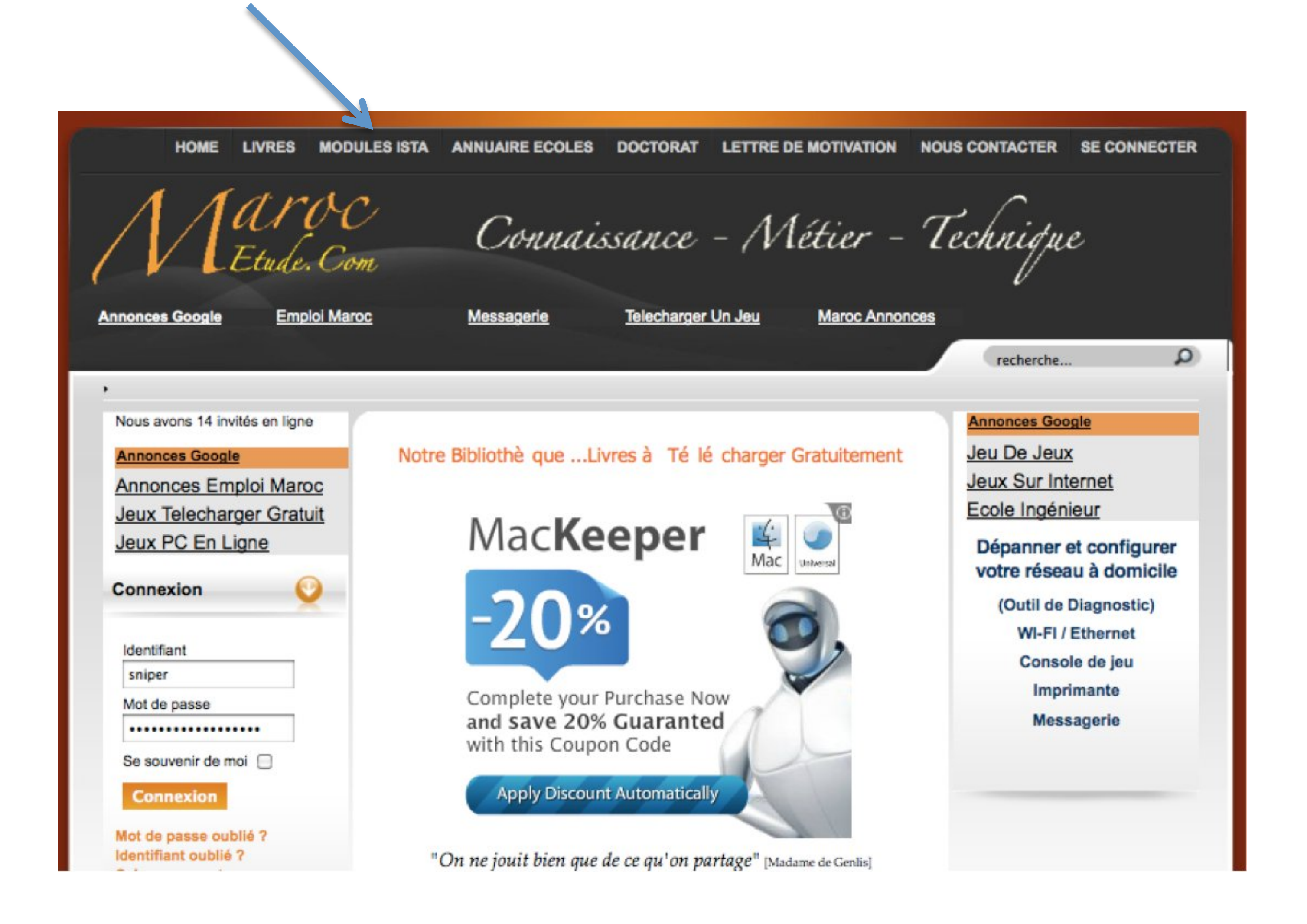
JUILLET 2005

PORTAIL DE LA FORMATION PROFESSIONNELLE AU MAROC

Télécharger tous les modules de toutes les filières de l'OFPPT sur le site dédié à la formation professionnelle au Maroc : **www.marocetude.com**

Pour cela visiter notre site www.marocetude.com et choisissez la rubrique :

MODULES ISTA



Document élaboré par :
Mr LAZAR BOTESCU : Formateur Animateur CDC/GE

<u>Pour la vérification et adaptation :</u>

Mme Natova Bisserka : Formatrice Animatrice CDC /FGT

SOMMAIRE

MODULE : N°17 INITIATION AUX AUTOMATES PROGRAMMABLES

Durée : 60 H

Théorique 30% : 18 H

Pratique 70% : 42 H

OBJECTIF OPERATIONNEL DE PREMIER NIVEAU DE COMPORTEMENT

COMPORTEMENT ATTENDU

Pour démontrer sa compétence le stagiaire doit utiliser un automate programmable selon les conditions, les critères et les précisions qui suivent.

CONDITIONS D'EVALUATION

- *À partir de directives.*
- *À l'aide :*
  - *de fiches techniques et du manuel d'utilisation d'un automate;*
  - *d'un programme en langage Grafcet ou en diagramme à échelons;*
- *Sur un automate programmable avec E/S " tout ou rien ".*

CRITERES GENERAUX DE PERFORMANCE

- *Utilisation appropriée de l'équipement informatique*
- *Respect des méthodes et des conventions de programmation d'un automate.*
- *Respect des normes.*

OBJECTIF OPERATIONNEL DE PREMIER NIVEAU DE COMPORTEMENT

PRECISIONS SUR LE COMPORTEMENT ATTENDU	CRITERES PARTICULIERS DE PERFORMANCE
A. Raccorder un automate.	• Repérage de l'information pertinence dans la documentation technique. • Localisation précise des points de raccordement • Câblage conforme au schéma de raccordement. • Reconnaissance précise des composants associés au matériel et des composants associés au logiciel.
B. Accéder aux fonctions d'un automate.	• Configuration précise de l'automate. • Utilisation appropriée du logiciel ou de la console de programmation dédiée. • Détermination juste du mode d'adressage.
C. Déceler des problèmes de fonctionnement d'un automatisme simple commandé par un automate.	• Respect de la procédure d'interrogation des E/S et des cases mémoire. • Indication juste des problèmes de fonctionnement.
D. Apporter des modifications mineures au programme d'un automate	• Modification conforme à la demande. • Respect de la procédure d'éducation. • Programmation précise des ajouts ou des retraits. • Respect de la procédure de sauvegarde
E. Effectuer l'essai d'un automatisme simple commandé par un automate	• Programmation fonctionnelle en simulation. • Système fonctionnel à la suite de la modification.

OBJECTIFS OPERATIONNELS DE SECOND NIVEAU

Le stagiaire doit maîtriser les savoirs, savoir-faire, savoir percevoir ou savoir être jugés préalables aux apprentissages directement requis pour l'atteinte de l'objectif de premier niveau, tels que :

Avant d'apprendre à raccorder un automate (A) :

1. *Décrire l'architecture d'un automate.*
2. *Décrire les applications d'un automate.*
3. *Reconnaître les différents modules E/S.*
4. *Repérer un logiciel de programmation.*

Avant d'apprendre à accéder aux fonctions d'un automate (B) :

5. *Distinguer les langages de programmation d'un automate*
6. *Décrire les principales instructions d'un automate.*
7. *Utiliser un logiciel de programmation.*

Avant d'apprendre à déceler des problèmes de fonctionnement d'un automatisme simple commandé par un automate (C) :

8. *Reconnaître l'état logique des E/S dans un programme par le mode dynamique (en ligne).*

Avant d'apprendre à effectuer l'essai d'un automatisme simple commandé par un automate (D) :

9. *Reconnaître les dangers potentiels.*

PRESENTATION DU MODULE

A titre indicatif :

Cette présentation doit :

- *Situer le module par rapport au programme de formation;*

Ce module doit être déroulé au Deuxième semestre de la deuxième année de formation.

- *Donner une description sommaire des grandes étapes de déroulement des activités d'apprentissage concernant la compétence visée par le module;*

- *Préciser la durée du module et les volumes horaires alloués aux parties théorique et pratique.*

Module : N° 17
AUTOMATE PROGRAMMABLE
RESUME THEORIQUE

Le contenu du résumé théorique doit couvrir l'ensemble des objectifs visés par la compétence relative au module en question en développant :

- *Des concepts théoriques de base (Définition, schémas illustratifs, démonstrations…..) ;*
- *Des exercices d'application ;*
- *Des évaluations (Contrôles continus).*

1. INTRODUCTION

Un automate programmable permet de remplacer une réalisation câblée comportant des composants combinatoires (portes) et séquentiels (bascules, séquenceurs,...) par un programme. Un programme est une suite d'instructions, qui sont exécutées l'une après l'autre. Si une entrée change alors qu'on ne se trouve pas sur l'instruction qui la traite et que l'on ne repasse plus sur ces instructions, la sortie n'est pas modifiée. C'est la raison de la nécessité de bouclage permanent sur l'ensemble du programme.

Par rapport à un câblage, on a donc deux désavantages : temps de réponse (un changement des entrées sera pris en compte au maximum après le temps d'un passage sur l'ensemble du programme, c'est ce qu'on appelle le temps de scrutation, qui sera souvent de l'ordre de la milliseconde), et non simultanéité (on n'effectue qu'un instruction à la fois). Mais ces temps étant en général très inférieurs aux temps de réaction des capteurs et actionneurs (inertie d'un moteur par exemple), ceci n'est que rarement gênant. L'avantage est que c'est programmable, donc facilement modifiable.

Tout automate programmable possède :

- *des entrées, des sorties, des mémoires internes : toutes sont binaires (0 ou 1), on peut les lire (c.a.d connaître leur état) (même les sorties), mais on ne peut écrire (modifier l'état) que sur les sorties et les mémoires internes. Les mémoires internes servent pour stocker des résultats temporaires, et s'en resservir plus tard.*
- *des fonctions combinatoires : ET, OU, NON (mais aussi quelquefois XOR, NAND,...)*
- *des fonctions séquentielles : bascules RS (ou du moins Set et Reset des bascules), temporisations, compteurs/décompteurs mais aussi quelquefois registres à décalage, etc...*
- *des fonctions algorithmiques : sauts (vers l'avant mais aussi quelquefois saut généralisés), boucles, instructions conditionnelles...*
- *de plus il permet de créer, essayer, modifier, sauver un programme, quelquefois par l'intermédiaire d'une console séparable et utilisable pour plusieurs automates. Désormais cette fonctionnalité est également possible sur PC, permettant une plus grande souplesse, une assistance automatique, des simulations graphiques,... mais pour un prix supérieur.*

Ce qui différencie les automates, c'est la capacité (entrées, sorties, mémoires internes, taille de programme, nombre de compteurs, nombre de temporisations), la vitesse mais surtout son adaptabilité (possibilité d'augmenter les capacités, de prendre en compte de l'analogique et numérique, de converser via un réseau...)

La schématisation, sous diverses formes, est un support de la communication technique. Elle est utilisée à diverses étapes du cycle de vie d'un produit ou d'un système. Une fois le schéma conçu, il faut utiliser tous les composantes dont leur symbole a été utilise, pour réaliser ce schéma. On appel ça une réalisation câblée.

On rappelle ci-dessous quelque schéma de commande pour l'alimentation d'une charge électrique.

.

*Fonction **OUI** : La sortie est à l'état 1 si et seulement si l'entrée est à l'état 1*

Schéma	***Symbole***	***Table de vérité***	***Équation***
a S	a S	a S / 0 0 / 1 1	***S=a***

*Fonction **NON** : La sortie est à l'état 1 si et seulement si l'entrée n'est pas à 1*

Schéma	***Symbole***	***Table de vérité***	***Équation***
$\bar{a}$ S	a S	a S / 0 1 / 1 0	$S=\bar{a}$

*Fonction **ET** : La sortie est à l'état 1 si et seulement si toutes les entrées sont à l'état 1.*

Schéma	***Symbole***	***Table de vérité***	***Équation***
a b S	a b S	b a S / 0 0 0 / 0 1 0 / 1 1 1 / 1 0 0	$S=a \cdot b$

*Fonction **OU** : La sortie est à l'état 1 si et seulement si, une ou plusieurs entrées sont à l'état 1.*

Schéma	***Symbole***	***Table de vérité***	***Équation***

Schéma	***Symbole***	***Table de vérité***	***Équation***
a, b, S	a, b, s	b a S 0 0 0 0 1 1 1 1 1 1 0 1	$S=a+b$

Fonction ***INHIBITION***

Schéma	***Symbole***	***Table de vérité***	***Équation***
a, $\bar{b}$, S	a, b, S	b a S 0 0 0 0 1 1 1 1 0 1 0 0	$S=a\cdot\bar{b}$

Fonction ***NON ET (NAND)***

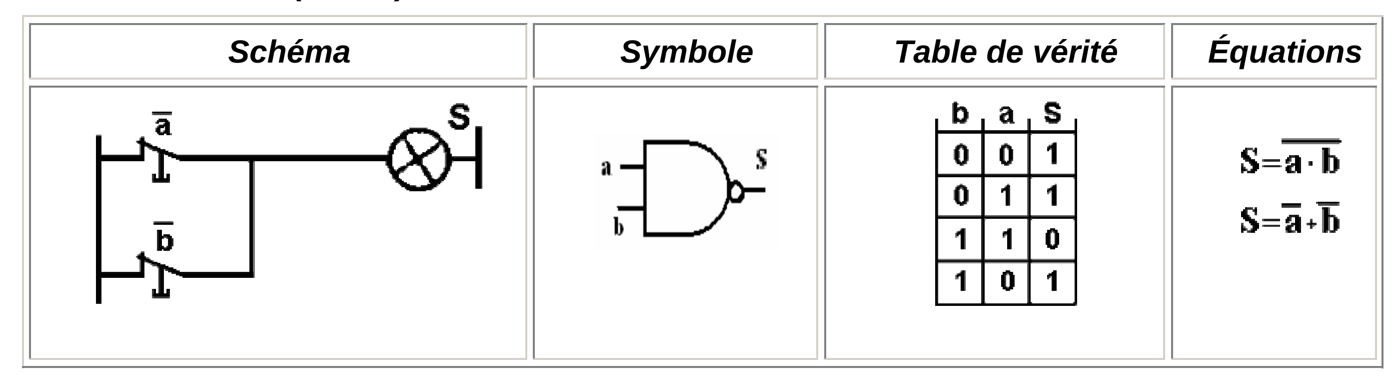


Schéma	***Symbole***	***Table de vérité***	***Équations***
$\bar{a}$, $\bar{b}$, S	a, b, S	b a S 0 0 1 0 1 1 1 1 0 1 0 1	$S=\overline{a\cdot b}$ $S=\bar{a}+\bar{b}$

Fonction ***NON OU (NOR)***

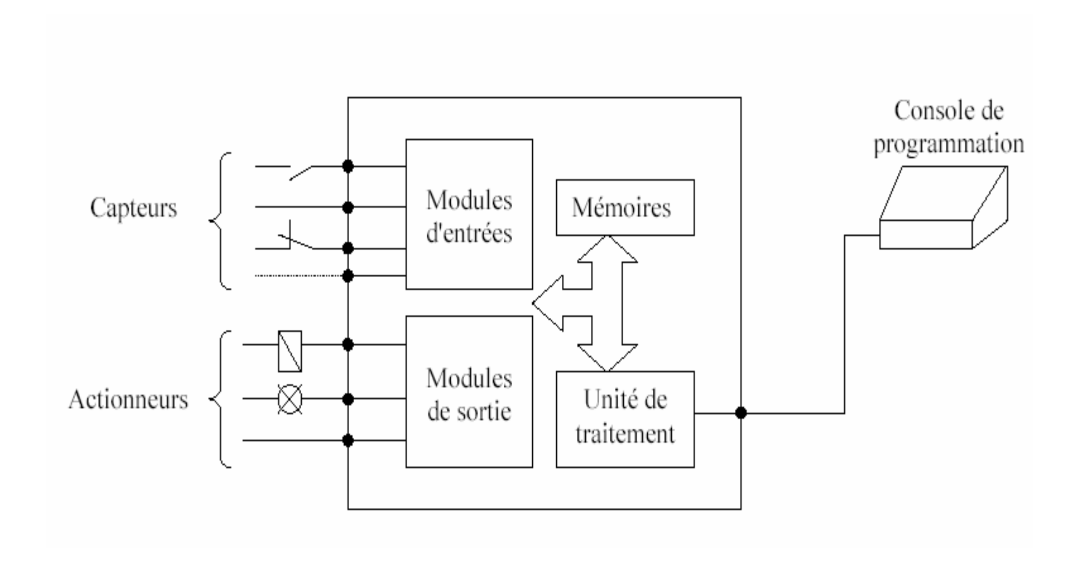
Schéma	***Symbole***	***Table de vérité***	***Équations***
$\bar{a}$, $\bar{b}$, S	a, b, S	b a S 0 0 1 0 1 0 1 1 0 1 0 0	$S=\overline{a+b}$ $S=\bar{a}\cdot\bar{b}$

REMARQUE : *Le schéma d'un installation électrique d'indifférent du quel degré de complexité n'est pas autre chose que une combinaison de circuits utilisant les schémas de bases antérieures.*

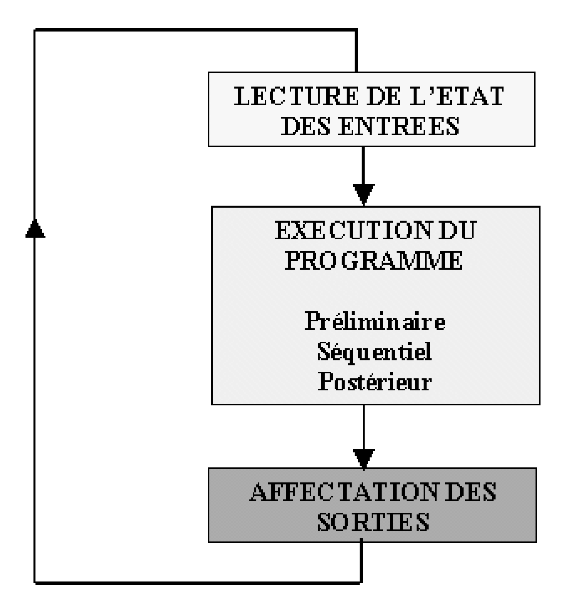
On peut distinguer deux grandes catégories des éléments constituant une installation automatisée :

- *des capteurs (qui fournissent des informations d'après l'état du système)*
- *des actionneurs (qui agissent auprès le système)*

La réalisation d'une schémas a base d'un automate programmable se reduit maintenant au couplage des capteurs aux entrées et des actionneurs aux sorties d'automate. Et après ça il ne reste que bien programmer l'automate !



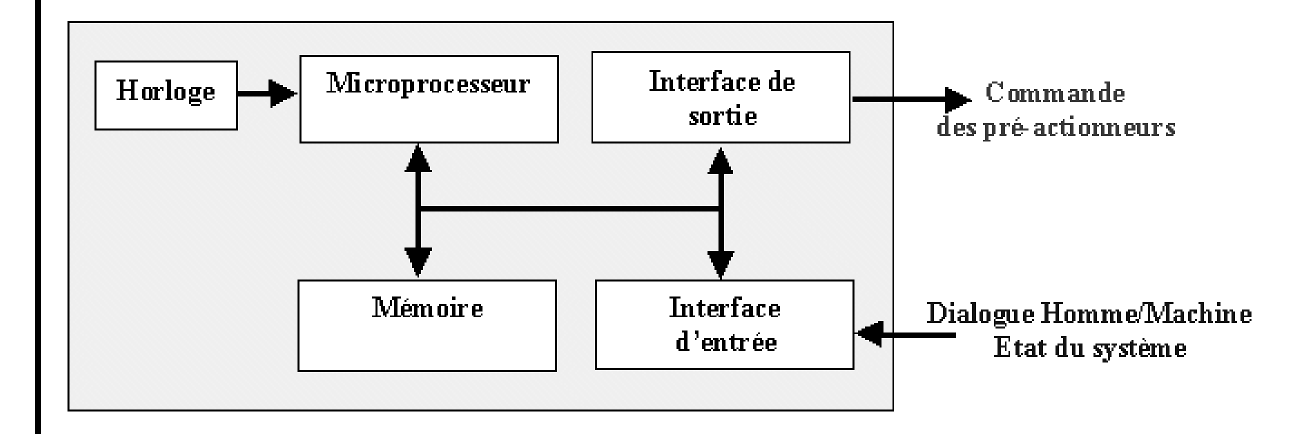
Pendant que l'automate se trouve en marche la suivante séquence se déroule continûment :



L'utilisation d'automate programmable est de plus en plus fréquente dans nos applications.

2. ARCHITECTURE D'UN AUTOMATE PROGRAMMABLE

La structure interne d'un API peut se représenter comme suit :



L'automate programmable reçoit les informations relatives à l'état du système et puis commande les pré-actionneurs suivant le programme inscrit dans sa mémoire.

Un API se compose donc de trois grandes parties :
- Le processeur ;
- La zone mémoire ;
- Les interfaces Entrées/Sorties

1)- Le microprocesseur :

Le microprocesseur réalise toutes les fonctions logiques ET, OU, les fonctions de temporisation, de comptage, de calcul... à partir d'un programme contenu dans sa mémoire. Il est connecté aux autres éléments (mémoire et interface E/S) par des liaisons parallèles appelées ' BUS ' qui véhiculent les informations sous forme binaire..

2)- La zone mémoires :

a)- La Zone mémoire va permettre :

- *De recevoir les informations issues des capteurs d'entrées*
- *De recevoir les informations générées par le processeur et destinées à la commande des sorties (valeur des compteurs, des temporisation, …)*
- *De recevoir et conserver le programmable du processus*

b)-Action possible sur une mémoire :

· ***ECRIRE*** *pour modifier le contenu d'un programme*
· ***EFFACER*** *pour faire disparaître les informations qui ne sont plus nécessaire*
· ***LIRE*** *pour en lire le contenu d'un programme sans le modifier*

Remarque :

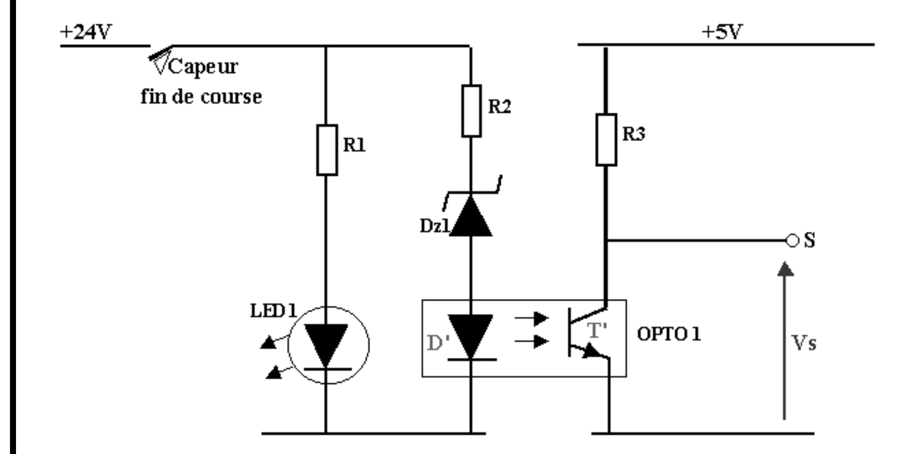
La capacité mémoire se donne en mots de 8 BITS (Binary Digits) ou octets.
Exemple:
Soit une mémoire de 8 Koctets = 8 x 1024 x 8 = 65 536 BITS. Cette mémoire peut contenir 65 536 informations binaires.

3) Les interfaces d'entrées/sorties

*Les **entrées** reçoivent des informations en provenance des **éléments de détection** et du **pupitre opérateur.***
*Les **sorties** transmettent des informations aux **pré-actionneurs** et aux **éléments de signalisation** du pupitre.*

a) Interfaces d'entrée digitales (ou TOR =tout ou rien)

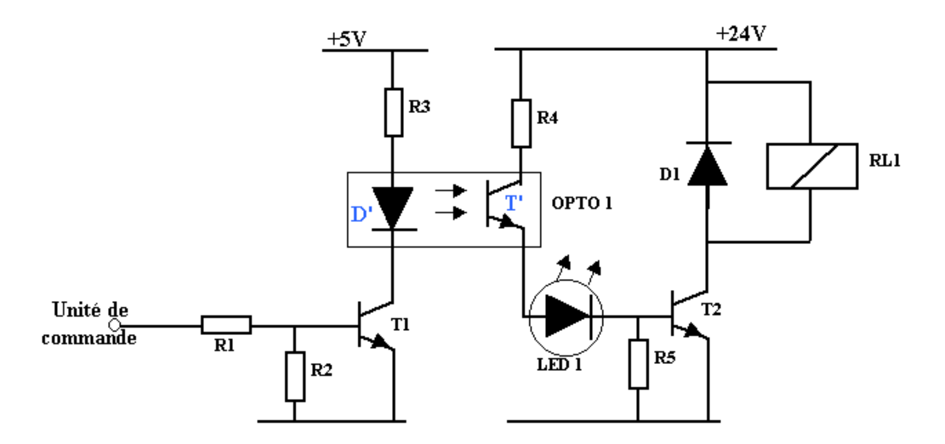
Elles sont destinées à :
- Recevoir l'information en provenance du capteur
- Traiter le signal en le mettant en forme, en éliminant les parasites et en isolant électriquement l'unité de commande de la partie opérative.



On remarque l'utilisation d'un optocoupleur qui assure une excellente isolation galvanique entre sa entrée et sa sortie (on peut dire « aucune contact électrique antre les deux cotés »)

b) Interfaces de sorties digitales (ou TOR = tout ou rien)

Elles sont destinées à :
- Commander les pré-actionneurs et éléments des signalisation du système
- Adapter les niveaux de tensions de l'unité de commande à celle de la partie opérative du système en garantissant une isolation galvanique entre ces dernières.



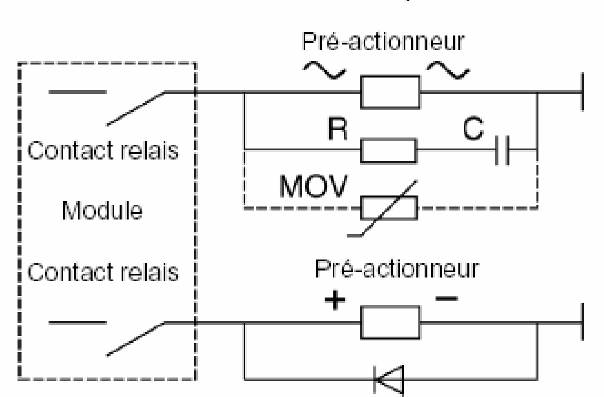
Protections des contacts des sorties à relais

Les contacts des sorties à relais n'intègrent pas de dispositif de protection afin de permettre la commande :

- d'entrées isolées galvaniquement, à faible niveau d'énergie et qui nécessitent l'absence de courants de fuite,
- de circuits de puissance en supprimant les surtensions inductives à la source.

De ce fait, il est obligatoire de monter aux bornes des bobines des pré-actionneurs :

- un circuit RC ou un écréteur MOV (ZNO) pour une utilisation en courant alternatif,
- une diode de décharge pour une utilisation en courant continu.



Note : Une sortie à relais utilisée sur une charge à courant alternatif ne doit pas être utilisée ensuite sur une charge à courant continu et vice et versa.

-

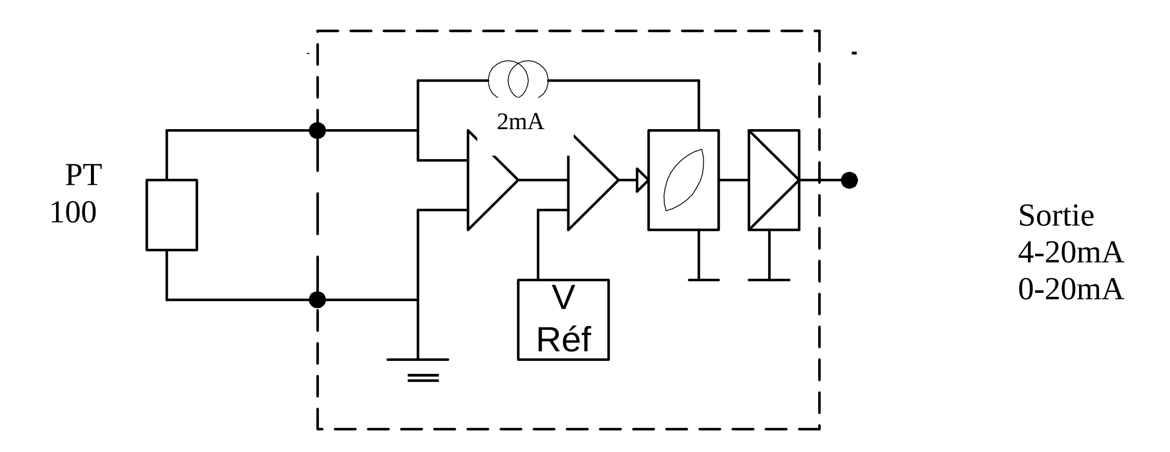
c) Interfaces d'entrée sortie analogiques

<u>Transmetteurs analogiques</u>

Les transmetteurs analogiques : Tension / intensité permettent d'adapter les signaux issus des capteurs pour les rendre compatibles avec l'unité de traitement. La variation de la grandeur d'entre est convertie en une variation :

- *En tension : de 0V, à10V*
- *En intensité : de 0 Ma à 20mA, ou de 4 mA à 20mA*

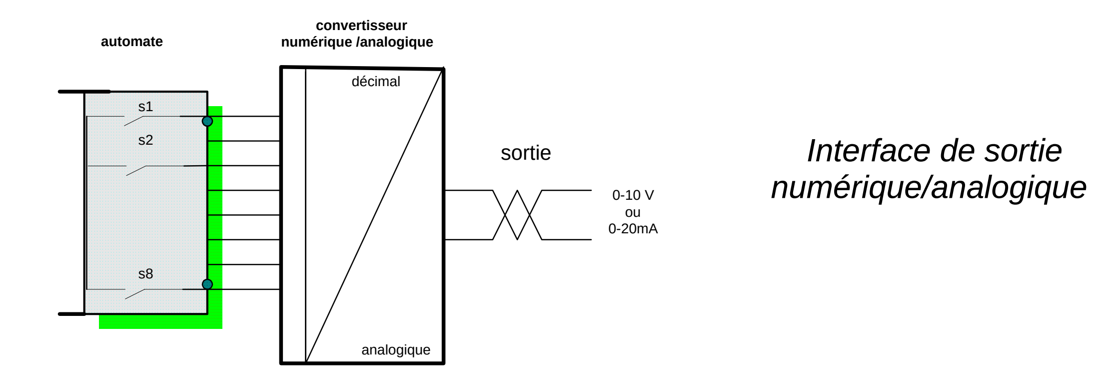
Exemple : Transmission de mesure de température effectuée par une sonde PT



Interface d'entrée analogique

d) Interfaces de sorties analogiques

Les conversions digitaux /analogiques ont pour fonction de générer un signal analogique normalisé (0-10 V ;0-20mA) à partir d' une information numérique, délivrée par l' unité de traitement et codée en binaire, sur des sorties digitales TOR raccordées aux entrées de l' interface(ou convertisseur).

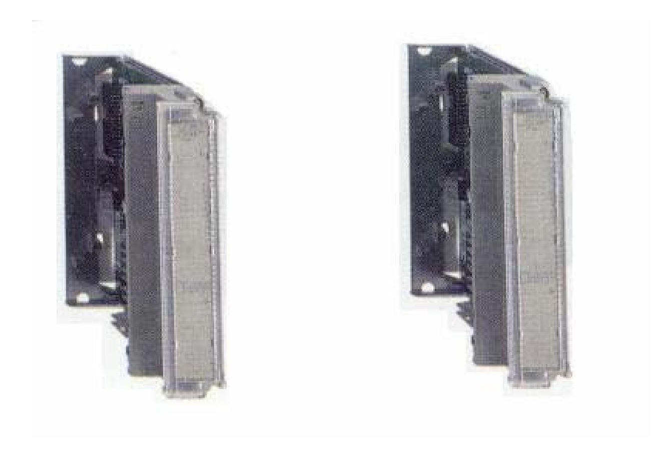


Interface de sortie numérique/analogique

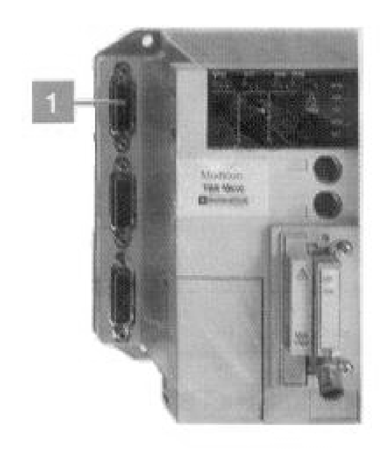
Quelques exemples pratiques :



Automate monobloc (les bornes sont des E/S)



Modules E/S d'un automate modulaire



Le n° 1 est un connecteur pour une entrée/sortie analogique (intensité et tension).

4) Alimentation de l'automate programmable industriel :

L'alimentation intégrée dans l'API, fournit à partir des tensions usuelles des réseaux (***230 V, 24 V=*** *) les tensions continues nécessaire au fonctionnement des circuits électroniques.*

5) Quelques types d'automates*.*

On présente dans le tableau suivante quelques constructeur d'automates et leurs produit

.

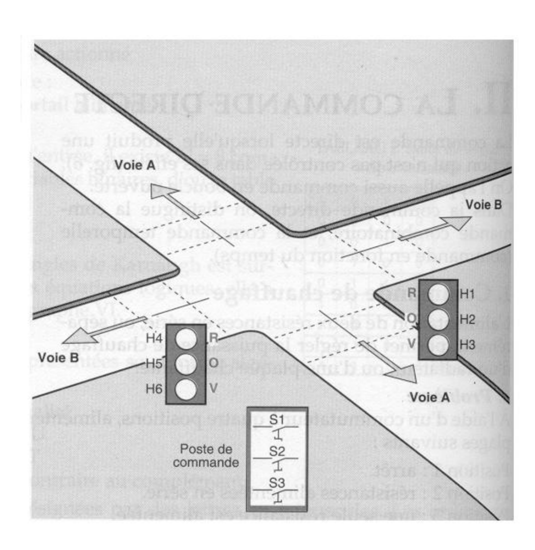
Marque	***Automate***	***Logiciel***
Télemecanique	*TSX Nano*	*Pl707*
	TSX 3708,TSx22	*Pl7-micro*
	TSX Premium	*Pl7 junior*
ALENBRADLEY	*SLC 500*	*APSF*
SIEMENS	*Serie5:S5* *Serie7:S7*	*Step 5* *Step 7*

Constructivement on peut classifie les automates dans deux grandes catégories:

a) automates monobloc
b) automates modulaires

3. EXEMPLES DES APLICATIONS UTILISANT UN API

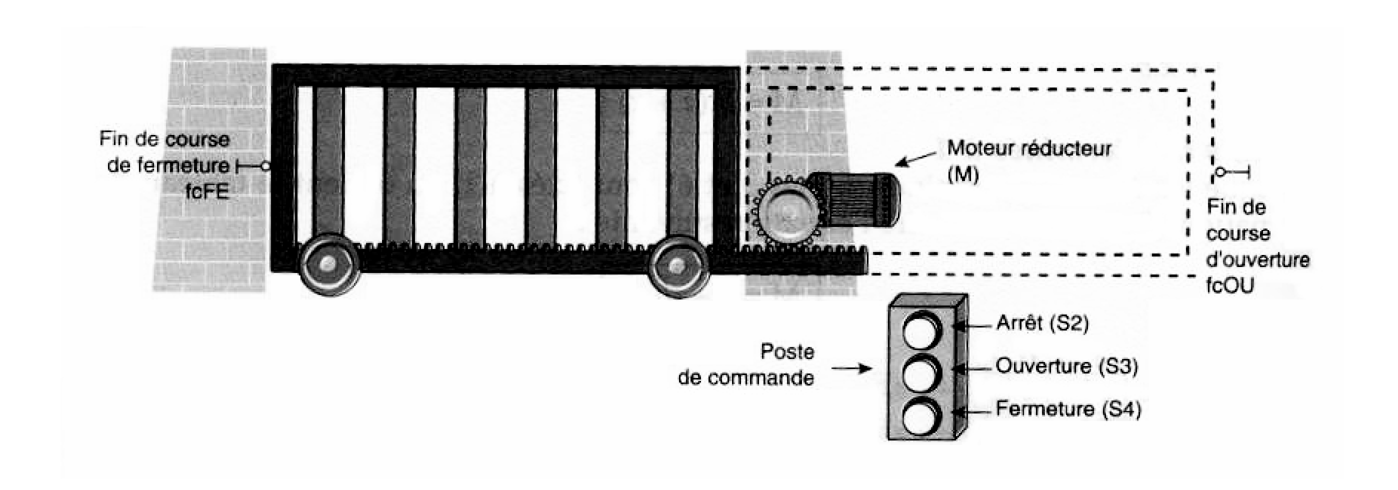
Exemple 1: Feux de carrefour



Description

On règle la circulation d'un carrefour de deux voies A et B par des feux tricolores (Rouges, orange, vert). Pour les circuits de commande classiques pour un tel type d'application on avait besoin des minuteries des circuits compliques d'interfaçage etc.,etc.
En utilisant un automate programmable les choses se simplifie au maximum. On établie le schéma de commande pour chaque feu, on fait le programme qui peut gérer cet fonctionnement, et ça y est !

***Exemple 2 : Portail coulissant**.*



Soit un portail coulissant à commander :

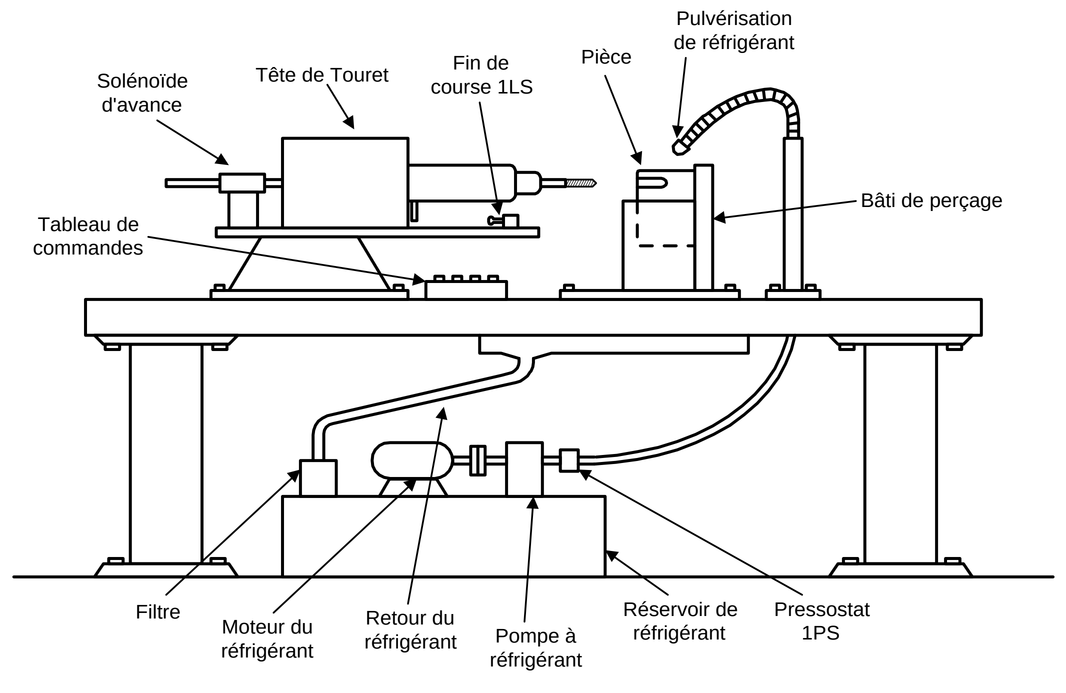
- *Le portail étant fermé, le contact fin de course fcFE est actionné ;*

- *On appuie sur le bouton-poussoir d'ouverture S3, le moteur actionne le portail et provoque son ouverture ;*
- *En fin d'ouverture, le contact fin de course fcOU est actionné, il signale l'ouverture du portail, et il coupe l'alimentation du moteur.*

L'action sur le bouton-poussoir de fermeture provoque l'inversion de sens de marche du moteur, et la fermeture du portail.

Le portail libère le contact fcOU, et se déplace jusqu'à actionner le contact fcFE qui provoque l'arrêt du moteur

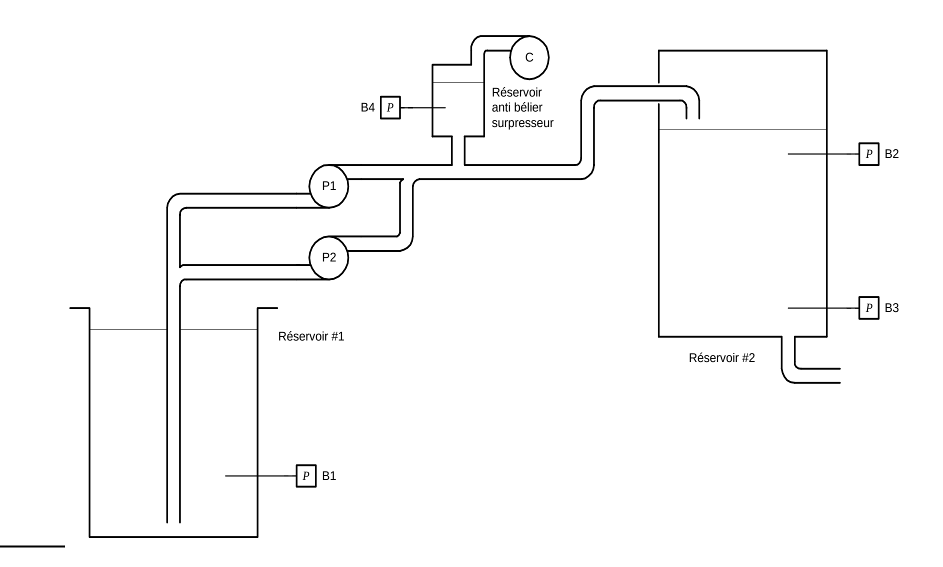
Exemple 3 : Système de perçage



Etapes :

- *On établie premièrement quels sont les actionneurs a commander par l'automate.*
- *Pour chacun de ceux-ci on définie les limites du déplacement/fonctionnement, donc on alloue les entrées provenant des capteurs (fin de cours, pressostats, etc.).*
- *On élabore le programme qui peut gérer le fonctionnement désiré de l'outillage.*

Exemple 4 : Système de pompage

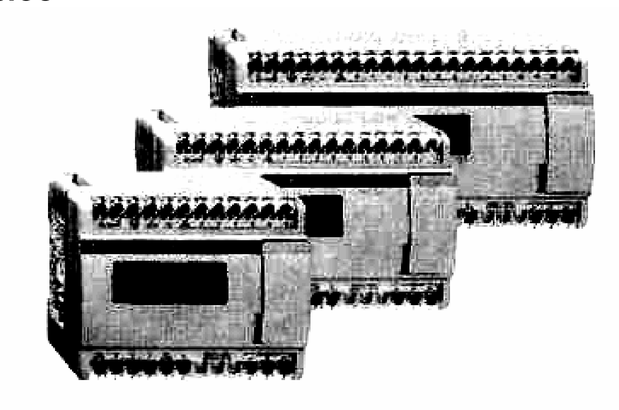


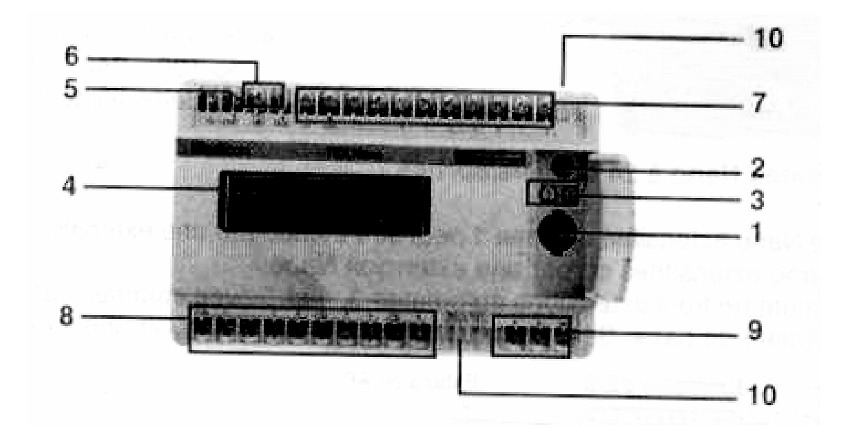
On passe par les mêmes étapes que dans les exemples antérieures pour définir les éléments a commander par les sorties d'automate (les actionneurs) , les informations provenant des capteurs pour les coupler aux entrées, et finalement l'élaboration du programme.

4. DESCRIPTION DES AUTOMATES

On distingue généralement deux types :les automates monoblocs et les automates modulaires.

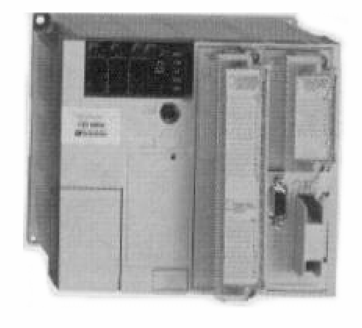
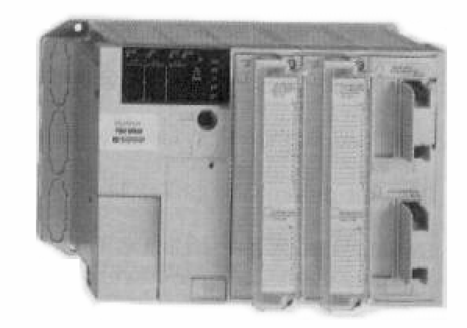
a) Automate monobloc

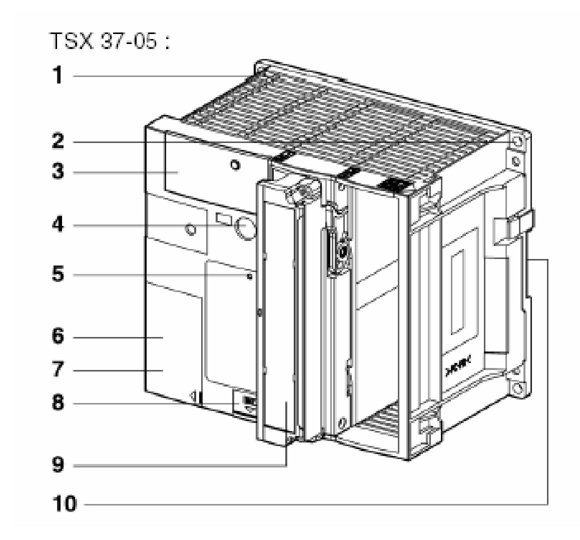




1- Une prise (1) pour raccordement du terminal de programmation .
2- Un sélecteur pour codage de la fonction base / extension.
3- Deux points de réglage analogique.
4- Une visualisation :
   - Des entrées 0 à 8 ou 0 à 13 et sorties 0 à 6 ou 0 à 9,
   - De l'état automate (RUN, ERR, COM, I/O).
5- Un raccordement de l'alimentation secteur
6- Une alimentation capteurs (=24V/150mA) sur modèles alimentés en ~100…240V.
7- Un raccordement des capteurs d'entrées.
8- Un raccordement des préactionneurs de sorties.
9- Un raccordement extension (extension d'entrées /sorties et / ou extension automate) ou raccordement Modbus esclave
10- Un cache amovible pour protection des borniers à vis.

b) Automate modulaire

1 Bac à 2 emplacements, intégrant l'alimentation, le processeur et sa mémoire.
2 Trou de fixation de l'automate.
3 Bloc de visualisation centralisée.
4 Prise terminal (TER).
5 Bouton RESET.
6 Trappe d'accès aux bornes d'alimentation.
7 Etiquette à renseigner pour le changement de la pile.
8 Trappe d'accès à la pile optionnelle et au commutateur de protection en écriture du système d'exploitation
9 Un module 28 E/S, positionné de base dans le premier emplacement.
10 Dispositif de montage sur profilé DIN.

*Les **AUTOMATES MONOBLOC** se caractérise par une réalisation compacte, c.a.d. un boîtier reliant tous les élément constituant : alimentation, interfaces d'entrée/sortie. Le nombre des entrées/sorties étant fixe, il faut choisir l'automate selon l'application concrète (nombre et type d'entrées/sorties, complexité du programme préconise), les changements ultérieures dans la conception de l'installation pouvant rendre inutilisable l'automate choisie initialement. Quand même, les uns des constructeurs ont prévu des solutions pour coupler plusieurs automates en vue d'élargir les performances initiales de l'installation.*

*Par contre les **AUTOMATES MODULAIRES** se caractérise par une famille des modules pour un type d'automate, l'utilisateur n'ayant qu'a choisir les modules qui en étant rallier ensemble peuvent satisfaire les demandes de l'application préconisée. Les changement ultérieures dans la structure et la complexité de l'installation ne présent aucun problème pour ce type d'automates. Si besoin, on peut ajouté des modules à la structure initiale d'une telle manière que les demandes soit satisfaites. Le nombre total des entrées/sorties peut arrivé pour ceux types d'automates a quelques centaines ou même milliers !*

c) Bloc de visualisation

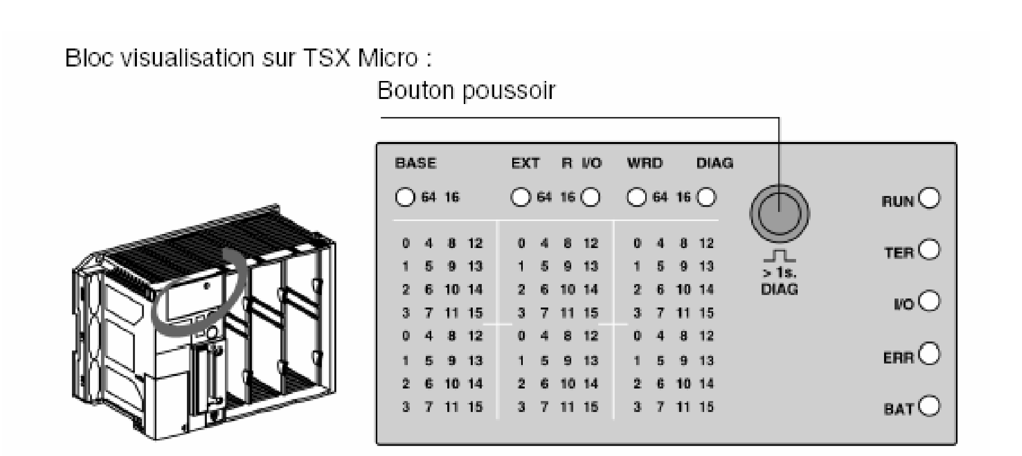
Le bloc de visualisation propose de manière centralisée, un ensemble de services nécessaires à la mise en oeuvre, à l'exploitation, au diagnostic et à la maintenance de l'automate, de tous ses modules positionnés dans le bac de base ou dans le mini-bac d'extension et des entrées/sorties TOR distantes :

- visualisation de l'état automate,
- visualisation de l'état des entrées/sorties locales ou distantes,
- test du câblage des entrées/sorties TOR, en l'absence de programme application,
- diagnostic des entrées/sorties et des modules,
- visualisation de données internes au programme (bits, mots, bits d'entrées/ sorties à distance,...).

Description

Le bloc de visualisation propose :

- sur la droite, 5 voyants d'état qui renseignent sur le fonctionnement de l'automate (RUN, TER, I/O, ERR, BAT),
- en partie supérieure, 5 voyants d'état qui renseignent sur le mode de visualisation en cours :
*- voyant **BASE** : mode visualisation des entrées/sorties de la base,*
*- voyant **EXT** : mode visualisation des entrées/sorties du mini bac d'extension,*
*- voyant **R I/O** : mode visualisation des entrées/sorties sur bus AS-i,*
*- voyant **WRD** : mode visualisation des objets du langage,*
*- voyant **DIAG** : mode diagnostic,*
- 3 blocs de 32 voyants qui renseignent sur les modules contenus dans l'automate ou dans son extension : état des entrées/sorties TOR, voies ou modules en défaut. De plus, chaque bloc est complété par 2 voyants par emplacement ("64" et "16") qui permettent de visualiser en deux fois, les modules 64 voies (16 premières voies d'entrées et 16 premières voies de sorties, puis 16 voie d'entrées/sorties suivantes),
- un bouton poussoir qui permet de visualiser la suite des informations et/ou de changer de mode de visualisation (mode visualisation des entrées/sorties ou diagnostic). En mode WORD, ce bouton poussoir permet de choisir la table des objets affichés.



d) Visualisation de l'état automate

La visualisation s'effectue au travers des 5 voyants RUN, TER, I/O, ERR et BAT qui renseignent par leur état (voyant éteint, clignotant ou allumé) sur le mode de fonctionnement de l'automate.

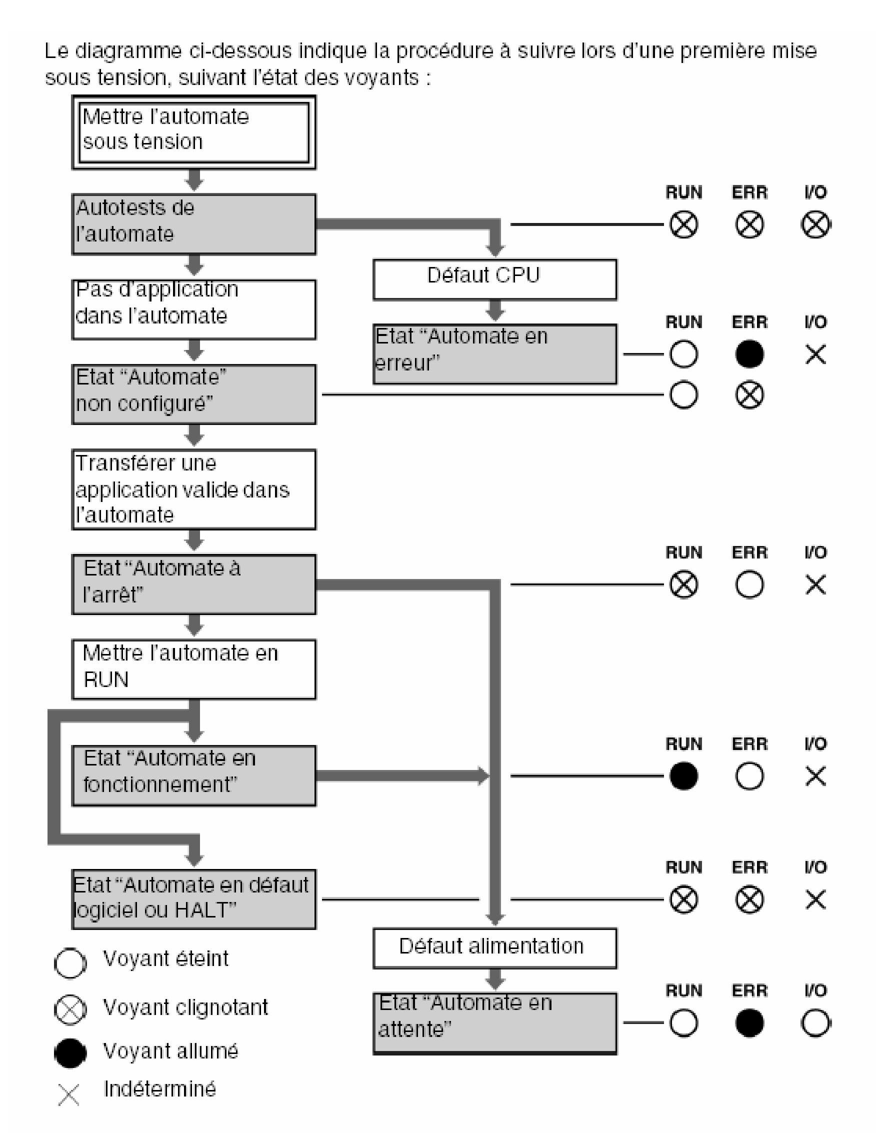
Descriptif ce tableau décrit l'état de l'automate en fonction des voyants :

Voyant	Etat
RUN	*ce voyant (de couleur verte) est allumé pour signaler que l'automate est en fonctionnement (RUN) et clignote pour indiquer qu'il est en STOP. Ce voyant est éteint lorsqu'il n'y a pas d'application valide dans l'automate ou lorsque celui-ci est en défaut.*
TER	*ce voyant (de couleur jaune) est allumé pour signaler que des informations sont échangées par la liaison terminal. Le trafic par la prise terminal peut donner l'impression que ce voyant clignote.*
I/O	*ce voyant (de couleur rouge) est allumé pour signaler un défaut relatif aux entrées/sorties :* *- défaut d'alimentation ou disjonction d'au moins une voie,* *- module absent, non conforme à la configuration ou hors service. Pour plus d'information sur les défauts signalés par le voyant I/O (défauts voie ou module), il est nécessaire d'appuyer plus d'une seconde sur le bouton poussoir pour passer en mode diagnostic (Voir Visualisation des entrées/sorties distantes sur le bus).*
ERR	*ce voyant (de couleur rouge) est allumé pour signaler un "défaut CPU" de l'automate. Ce voyant clignote lorsqu'il n'y a pas d'application valide dans l'automate ou lors d'un "défaut bloquant (Voir Recherche des défauts à partir des voyants d'état de l'automate, p. 204)" du programme application.*
BAT	*ce voyant (de couleur rouge) est allumé pour signaler la défectuosité ou l'absence de la pile (optionnelle). Cette pile qui assure la sauvegarde de la mémoire RAM nécessite d'être changée suivant la procédure (voir Mise en place/changement de la pile. Si le bit système %S66 est à l'état 1, l'allumage de ce voyant est inhibé.*

Récapitulatif

Illustration :

RUN — Marche/Arrêt de l'automate
TER — Trafic sur la prise terminal
I/O — Défaut d'entrées/sorties (voie ou module)
ERR — Défaut du processeur ou de l'application
BAT — Défaut ou absence de la pile

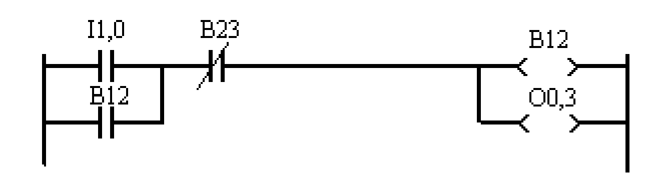


5. LE LANGAGE A CONTACTS DU TSX

C'est le langage de base des TSX. Il est nécessaire de le connaître même pour utiliser le langage PL7-2 (proche du Grafcet).

Sur le TSX, les sorties sont appelées Oi,0 à Oi,23 (i=numéro de carte d'entrée), les entrées Ii,0 à Ii,24. Les variables internes sont notées en décimal de B0 à B255 (B pour Bit interne ou Bobine).

La programmation se fait à l'aide de programmes graphiques : les réseaux. Ce sont des schémas qui sont exécutés l'un après l'autre, de haut en bas (et non suivant leur label). Chaque réseau est scruté par colonne de gauche à droite.
exemple :



Dans ce cas l'entrée B12 est l'ancienne valeur du bit interne (bobine) B12. Si l'on veut utiliser le résultat B12 de ce réseau, il faut utiliser B12 dans le réseau suivant.

On note un capteur par le signe, un contact complémenté (vrai si 0) par

Un bit interne est notée, un bit interne inverse (commandée par un niveau 0).

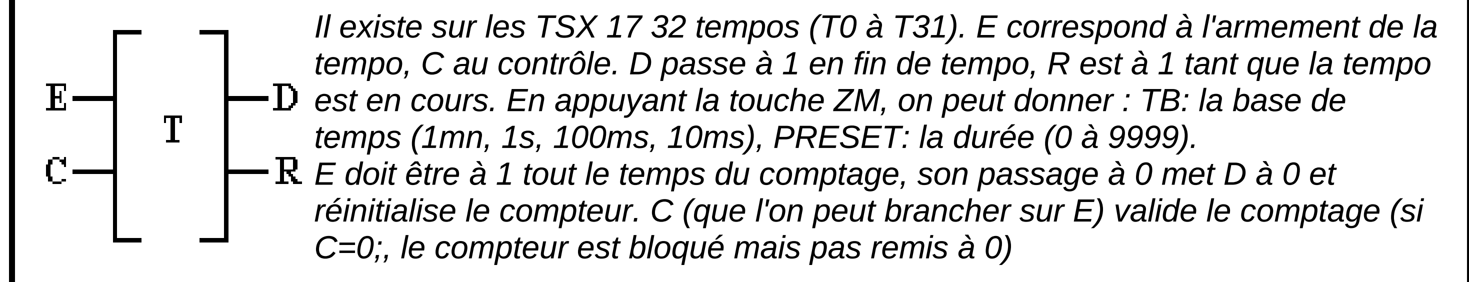
Une bascule bistable est allumée par S, éteinte par R.

Un saut à un autre réseau est noté J.

Un saut est effectué immédiatement lors de son évaluation (les bobines en sortie dans le même réseau mais sur les lignes suivantes ne seront donc pas évaluées en cas de saut). On a intérêt de n'utiliser que des sauts avants (vers la fin du programme). L'exécution du dernier réseau sera automatiquement suivie de l'exécution du premier (sauf si sauts). ***L'automate fixe automatiquement les entrées au début de cycle et n'affecte les sorties qu'en fin de cycle (les variables internes sont évidement immédiatement modifiées)****. Il est nécessaire de refaire un cycle (c'est à dire passer du dernier réseau au premier) fréquemment (tous les 150 ms maximum).*

Temporisation

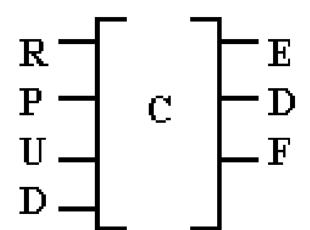
On représente la tempo par le signe :



Il existe sur les TSX 17 32 tempos (T0 à T31). E correspond à l'armement de la tempo, C au contrôle. D passe à 1 en fin de tempo, R est à 1 tant que la tempo est en cours. En appuyant la touche ZM, on peut donner : TB: la base de temps (1mn, 1s, 100ms, 10ms), PRESET: la durée (0 à 9999).
E doit être à 1 tout le temps du comptage, son passage à 0 met D à 0 et réinitialise le compteur. C (que l'on peut brancher sur E) valide le comptage (si C=0;, le compteur est bloqué mais pas remis à 0)

On dispose également de 8 tempos monostables M0 à M7, avec une seule entrée S, une seule sortie R valant 1 à durant le temps présélectionné, partir du front montant de S, indépendamment du moment de passage à 0 de S. Un nouveau front montant de S en cours de comptage relance le compteur à 0.

Compteur / décompteur



Il existe sur les TSX 17 32 compteurs (C0 à C31). R (reset) met le compteur et les sorties à 0. P (preset) met le compteur à la valeur finale et la sortie D (done) à 1 (sauf si R=1). U (up) incrémente le compteur, D (down) le décrémente. La sortie F (full) vaut 1 lors du passage du compteur (par U) de 9999 à 0, E (empty) lors du passage (par D) de 0 à 9999. Si U=D=1, le compteur est inchangé.

La valeur de préselection (Ci,P, entre 0 et 9999) se définit en "zoomant" sur le compteur.
Les autres fonctions disponibles (comparateurs, opérations arithmétiques et logiques, piles, registres à décalage, transcodage binaire, BCD, ASCII...) sont détaillées dans le chapitre 5 du document "Terminal TSX T407 Modes opératoires PL7-2", au chapitre 2 de "Langage PL7-2 Synthèse" ainsi qu'au chapitre B4 du manuel de référence du PL7-2.

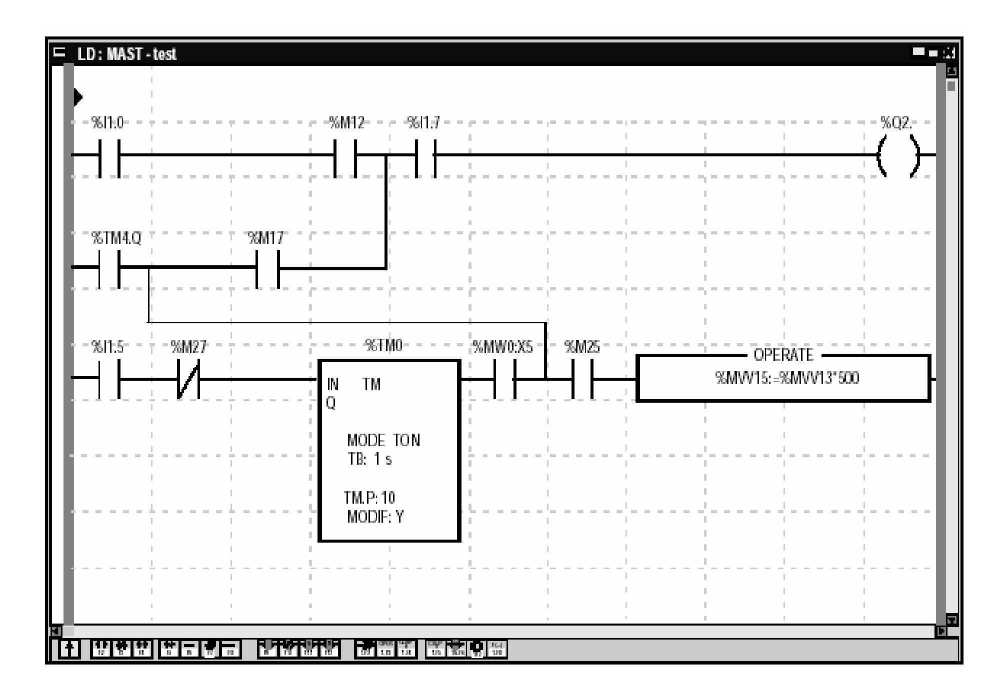
Conclusion

Ces fonctions de base (tempo, compteur) sont présentes dans tous les automates (même si elles sont mises en oeuvre par des langages très différents), sauf les sauts qui peuvent être plus limités (au minimum bouclage automatique sur l'ensemble du programme, mais sans sauts dans le programme). Mais le principe reste valable quel que soit l'automate. Souvent, d'autres possibilités existent, en particulier temporisations, comptage, comparaisons,...

Edition langage à contact

Présentation L'éditeur Ladder offre de nombreux outils assurant la construction des réseaux de contact de façon conviviale:
- *une palette d'éléments graphique,*
- *les objets du langage peuvent être indifféremment saisis et visualisés sous forme de repères, de symboles, ou les deux à la fois,*
- *une vue réduite.*

Editeur:

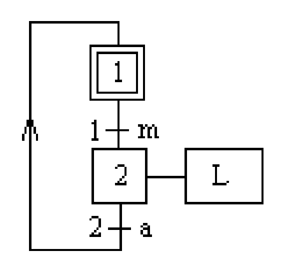


L'éditeur permet l'appel immédiat à des fonctions d'aide à la saisie:
- accès aux bibliothèques de fonctions,
- saisie des variables sous forme de symboles ou repéres.

En visualisation, les réseaux sont présentés sous forme contractée. Il est ainsi possible de visualiser plusieurs réseaux dans la même fenêtre, et d'y accéder par la barre de défilement ou par leur étiquette.
L'accès à un sous-programme peut s'effectuer directement à partir du programme d'appel.

6 PROGRAMMATION DIRECTE EN GRAFCET

Certains automates sont prévus pour être programmés plus facilement à partir d'un Grafcet. C'est le cas du TSX (à l'aide d'une cartouche ROM) mais pas du MICRO1.
Il faut tout d'abord dessiner le Grafcet. Analysons le cas du Grafcet suivant :

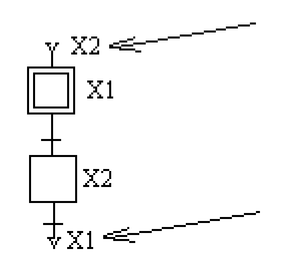


choix des adresses et variables internes

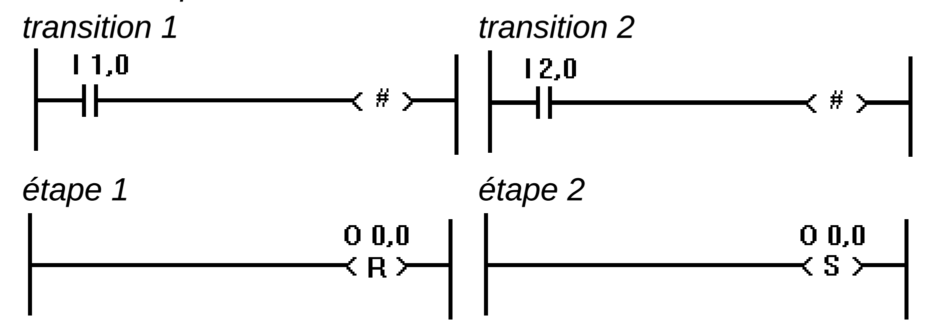
- *entrées :*
  - *m : entrée I1,0*
  - *a : entrée I2,0*
- *sortie L : O0,0*

programme :

On vient de l'étape 2
Les liaisons verticales vont de haut en bas uniquement. Par contre on peut remplacer une liaison par 2 flèches, en précisant l'étape d'où l'on vient et celle où l'on va. C'est ce que l'on utilise pour une remontée.
On va à l'étape 1



Une fois le grafcet entré, on définit les transitions et les actions correspondant aux étapes. Pour ceci, placer le curseur sur une transition à définir, appuyer la touche ZM (zoom). Un réseau à contacts apparaît, avec un bit interne représentant la transition. Il faut alors représenter le schéma qui, en fonction des capteurs, "allumera" la réceptivité. On valide le réseau par ENT (touche ENTER). Pour les actions, on peut (mais je ne le conseille pas) pointer une étape, appuyer ZM, donner le schéma qui allumera les bobines de sortie. Sur nos TSX, les sorties ne peuvent être activées que par un bit interne <S>, ce qui force à désactiver la sortie par un bit interne <R> l'étape suivante.
Soient : Capteurs m=I1,0, a=I2,0 Sortie L=O0,0. Les réseaux à entrer sont donc:

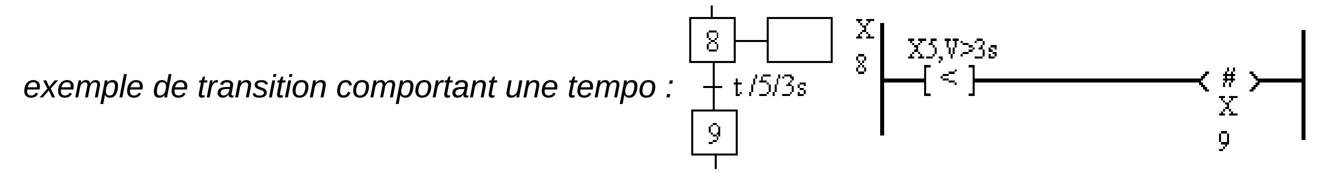


Une bien meilleure solution est de regrouper toutes les actions dans le "traitement postérieur". Attention, du fait que le TSX fige les entrées-sorties le temps d'un cycle, il ne faut mettre en place qu'une seule "équation" par sortie (sinon seule la dernière sera prise en compte). On n'oubliera donc pas de regrouper (en parallèle) les Xi allumant une sortie.

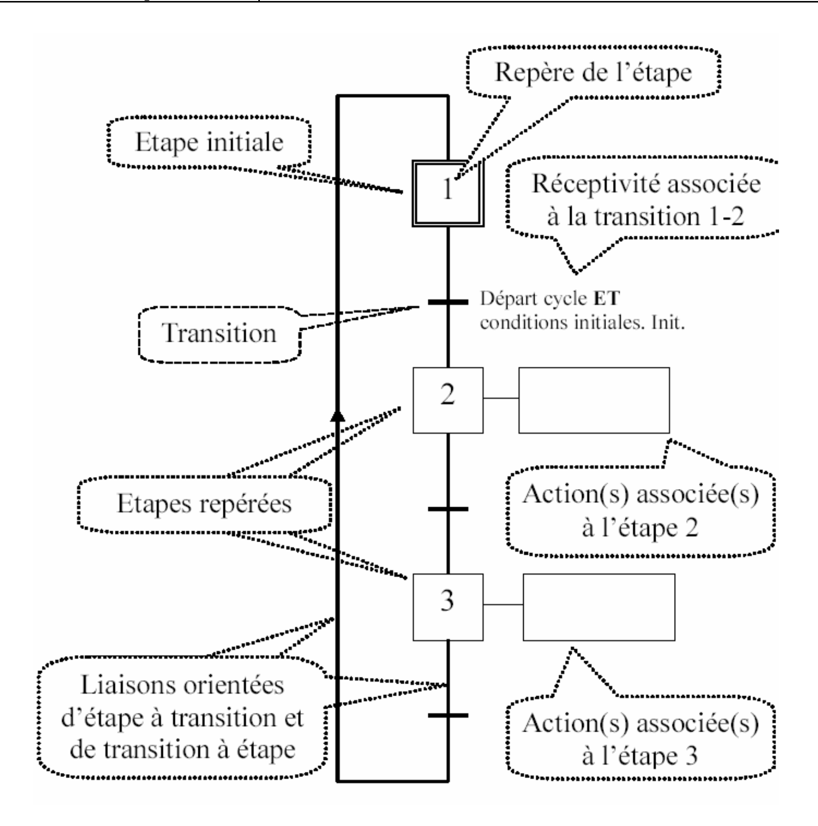
X2 — O 0,0

Une ***tempo*** *(en secondes) est automatiquement liée à chaque étape, et permet de tester la durée depuis l'activation de l'étape par un opérateur de type -[<]- (comparaison) par la variable interne Xi,V (i numéro d'étape)*

exemple de transition comportant une tempo :



Les éléments d'un grafcet sont donc :



7 UTILISATION DE LA CONSOLE DE PROGRAMMATION

L'operateur peut communiquer avec l'automate soit à travers un P.C portable, fixe (Fig.1) ou avec la console (Fig. 2). On lie l'automate au PC (ou à la console) par un câble (RS232)

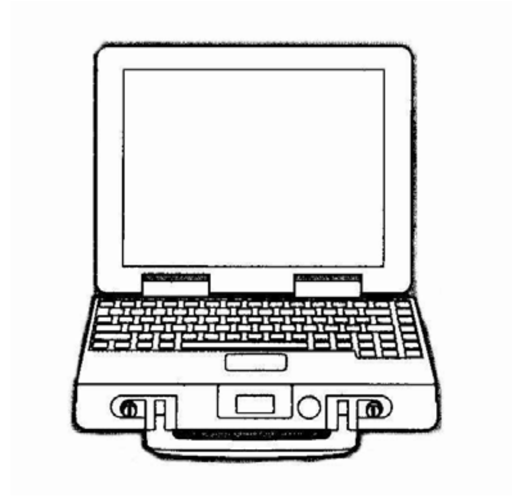


Fig.1 PC portable



Fig.2 La console

Ce chapitre précise l'utilisation des claviers et les branchements à effectuer.

Allumez l'automate puis la console (touche ON). Mettez l'automate en mode STOP (sur la platine portant l'automate). Quand le menu principal est affiché, choisissez PRG (programme) en appuyant la flèche qui se trouve sous cette option. Vous pouvez alors soit examiner le programme en mémoire, soit visualiser la suite des options du menu par la flèche sous le -/-. On choisit alors l'option CLM (clear memory), qui nous demande en quel langage on veut travailler (SEQ pour grafcet, LAD pour contacts). Puis après quelques secondes, on peut revenir au programme principal par la touche QUIT (qui remonte brutalement) ou 2 fois CLR (qui remonte avant la dernière commande effectuée).

Pour entrer le programme, choisir PRG puis SEQ ou LAD. On valide un réseau ou tout le programme par la touche ENT.

Pour faire tourner le programme, revenez au menu principal (QUIT), choisissez DBG (debug) puis R/S (run/stop), ou mettez en RUN par le contacteur RUN/STOP de la platine.. Quand le programme tourne, on peut revenir sous PRG pour le voir, mais pas le modifier. Les capteurs et les bobines sont alors représentées en pointillés si à 1, en trait plein si à 0.

Les modifications se font en revisualisant le programme sous PRG, choisir le bon reseau ou partie de Grafcet et choisir l'option MOD. En général, une modification nécessite d'effacer le capteur ou trait existant (touche SH + DEL) et remettre le nouveau,
Dans d'autres cas (traits horizontaux par exemple), on efface une entité en superposant une entité identique.

Un cycle de programme en langage Grafcet peut être précédé par un programme en langage à contacts (ladder) appelé traitement préliminaire, et suivi d'un traitement postérieur. La scrutation des entrées se faisant avant le traitement préliminaire, on peut y traiter des conditions sur les entrées préliminaires ou effectuer une partie d'un calcul au cas où un réseau ne suffirait pas pour une réceptivité. Le traitement postérieur se fait avant l'affectation des sorties, et peut donc modifier des sorties avant leur affectation définitive. Ces traitements peuvent utiliser l'état du Grafcet (par l'intermédiaire des bits Xi). A la mise en route de l'automate, tous les bits internes (sauf indication contraire) sont mùis à 0, sauf les étapes initiales.

Description des menus (utiles) sur la console T407

- *Menu principal [TSX 17-20]*
    - *ADJ (adjust) permet de visualiser ou modifier toute variable.*
    - *DBG (debug) : mise au point : permet de visualiser le programme et voir l'état des capteurs, sorties, étapes actives... (trait plein dans le programme si actif, interrompu si 0) et mettre des points d'arrêt dans le programme.*
    - *PRG : créer ou modifier le programme.*
    - *TRF (transfert) pour mémorisation sur EEPROM et impression sur imprimante (RS232).*
- *Menu PRG (dans tous les cas)*
    - *CLM (clear memory) efface le programme actuel, permet de définir si le nouveau programme sera en langage à contacts (LAD) ou Grafcet (SEQ).*

- *CNF (config) configuration de l'automate, de la liaison RS232 pour l'imprimante (LINE), des bobines à sauvegarder même en cas de coupure de courant (SAV)...*
- *NAME permet de donner un nom au programme.*
- *LK vérifie si le programme en mémoire ne comporte pas d'erreur.*
- *FREE retasse le programme (à faire après de nombreuses modifications).*

- *Menu PRG en mode ladder (LAD)*
  - *TOP aller au premier réseau*
  - *BOT (bottom) aller après le dernier réseau (on passe ensuite au dernier par la flèche vers le haut)*
  - *LAB : donner un numéro de réseau (label) puis [ENT] pour le visualiser*
  - *INS insère un nouveau réseau (vide) devant le réseau actuel.*
  - *DEL (delete) supprime le réseau actuel.*
  - *SCH (search) permet de rechercher tous les réseaux comportant une bobine ou contact donné.*
  - *[ZM] (zoom) visualise l'adresse d'un contact ou bobine (exemple I1,2), on peut se déplacer dans le réseau par les flèches.*
  - *[CLR] (clear) retour au niveau supérieur (ZM->LAD->PRG->principal)*
  - *[Quit] retour direct au menu principal.*
- *en mode ZOOM (sous PRG en mode LADDER)*
  - *LAB donner au réseau actuel un numéro de label (0 à 999)*
  - *" " donner un commentaire au réseau actuel (15 caractères maxi, sera affiché au dessus du réseau).*
  - *MOD permet de modifier l'élément pointé (l'effacer par [DEL] par exemple), on valide le réseau modifié par [ENT].*
- *Menu PRG en mode GRAFCET*

  On dispose de 8 pages (0 à 7) de 14 lignes de 8 colonnes. On peut au maximum prendre en compte 96 étapes, les divergences et convergences sont limitées à 4 voies. L'écran ne montre qu'une petite partie de la page, mais le numéro de page (P), de ligne (L) et de colonne (C) sont toujours affichés. On se déplace par les flèches, ou en tapant P, L, C ou X (étape) suivi du numéro désiré. Les fonctions sont approximativement les mêmes qu'en mode ladder, hormis :

  - *DLP : effacement d'une page complète*
  - *[ZM] face à une transition, la définit (si réseau vide, réceptivité toujours fausse)*

   - *[ZM] face à une étape, définit son action (étape d'attente si réseau vide)*
   - *MOVE : déplace l'élément actuel (par les flèches) puis valider par [ENT]*

- *Menu DBG*
   - *R/S passe de RUN à STOP et inversement (on peut aussi utiliser le contacteur sur la platine).*
   - *PRG : visualiser le programme et l'état des variables (trait plein=1, pointillé=0), insertion de points d'arrêt.*
   - *CY/ : exécution cycle par cycle*
   - *STP : liste des étapes actives*
   - */L point d'arrêt sur un label, /o sur une étape.*
   - *S/L et S/o : blocage sur un label ou une étape.*

---

8 EDITEUR LANGAGE GRAFCET

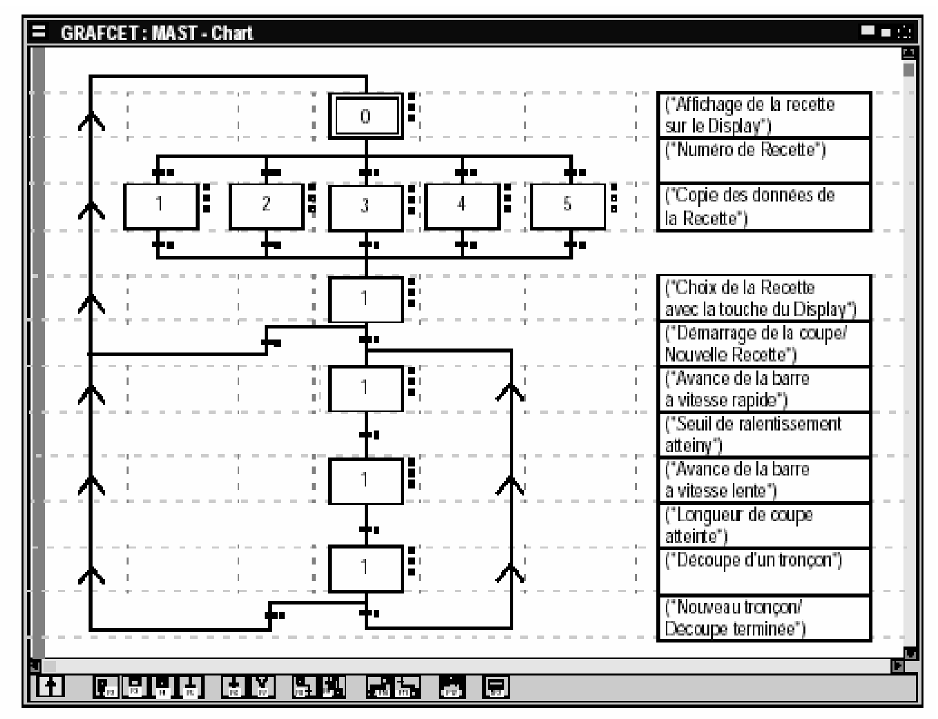
Présentation

L'éditeur dispose de nombreux outils permettant la saisie du graphe de façon conviviale:

- des palettes d'objets graphiques,
- un accès à la programmation des actions ou des réceptivités,
- une numérotation automatique des étapes,
- un affichage par page grafcet avec les lignes d'étapes et de transitions,
- une saisie simplifiée des commentaires,
- une vue réduite.

La construction du graphe s'effectue en sélectionnant l'objet désiré dans la palette graphique et en le plaçant dans la page grafcet.

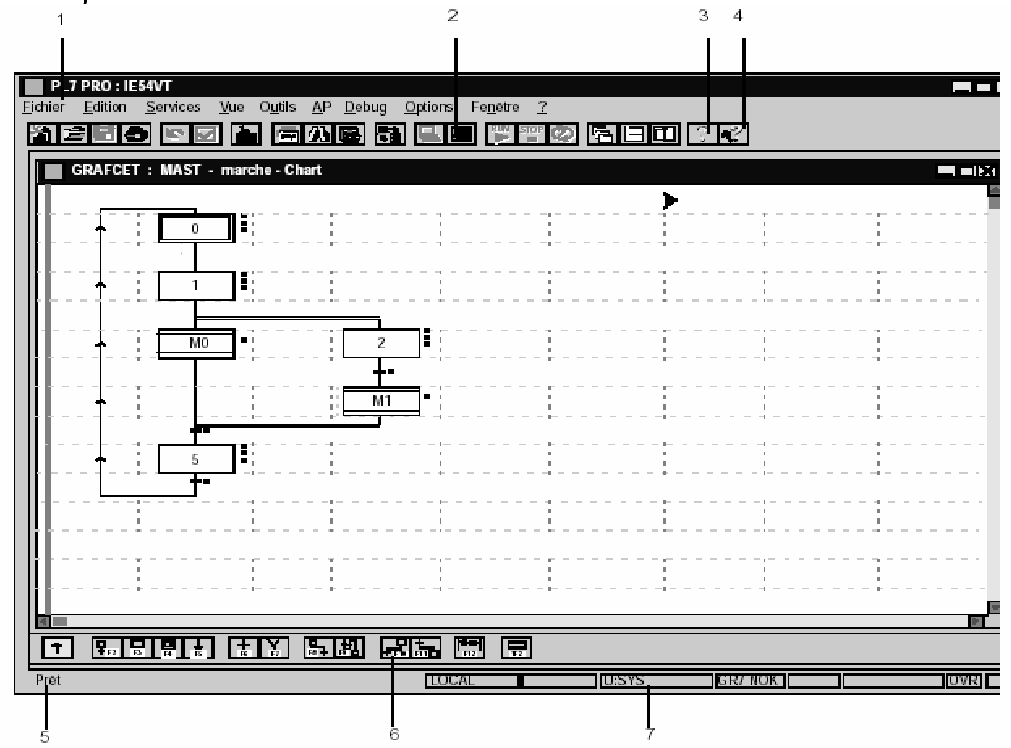
La visualisation immédiate des objets graphiques programmés, est assurée par une évolution de leur graphisme (traits fins).



L'éditeur grafcet se comporte comme une zone d'édition se déplaçant sur un module complet de 8 pages grafcet.

Eléments de base

Le logiciel PL7 utilise l'ergonomie de Windows et se présente de la manière suivante:Exemple de fenêtre:



Ce tableau donne la description des différentes zones:

Repère	Description
1	Barre de menu permettant l'accès à toutes les fonctions du logiciel.
2	Barre d'outils offrant un accès rapide par la souris à toutes les fonctions de base.
3	Aide en ligne sur l'utilisation du logiciel.
4	Aide contextuelle du logiciel.
5	Zone de commentaire.
6	Palette d'éléments graphiques.
7	Contexte de travail.

Barre d'outils La barre d'outil assure un accès rapide aux fonctions de base du logiciel:



Ce tableau donne la signification de chaque élément de la barre d'outils :

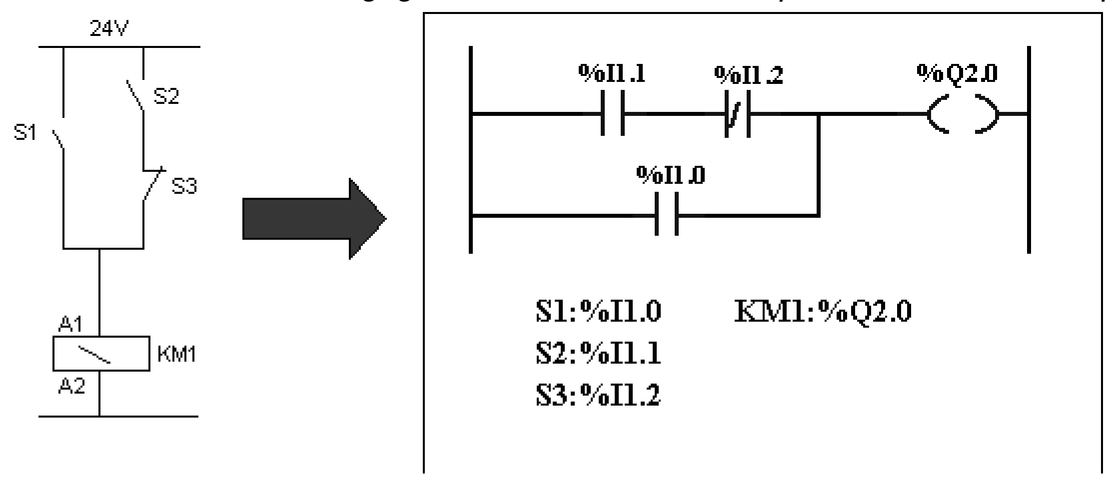
Elément	Fonction	Elément	Fonction
	Nouvelle application		Mode local
	Ouvrir une application		Mode connecté
	Enregistrer l'application		Passage de l'automate en RUN
	Imprimer tout ou partie de l'application		Passage de l'automate en STOP
	Annuler les dernières modifications		Lancer / Stopper l'animation
	Valider les modifications		Organisation des fenêtres en cascade
	Atteindre		Organisation des fenêtres en mosaïque horizontale
	Navigateur application		Organisation des fenêtres en mosaïque verticale
	Références croisées		Aide
	Bibliothèque de fonctions		Qu'est-ce que c'est ?
	Transfert automate <-> console		

9 PROGRAMMATION DU TSX MICRO EN LANGAGE PL7-MICRO

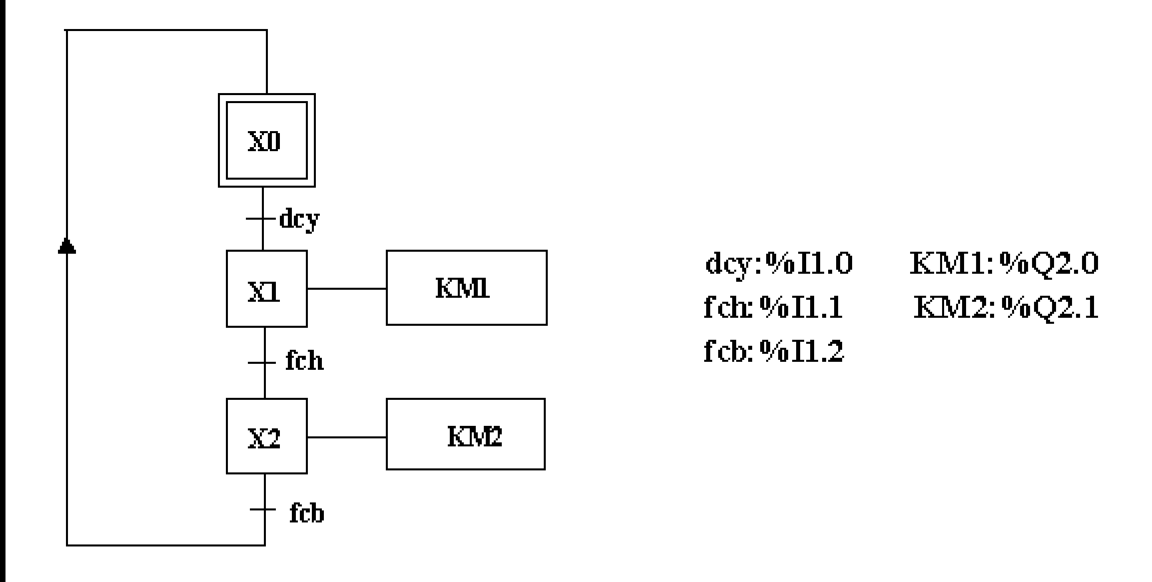
Le PL7-micro associe deux langages :
· Le Ladder " schéma à contacts "
· Le Grafcet " Chart "

Le ladder

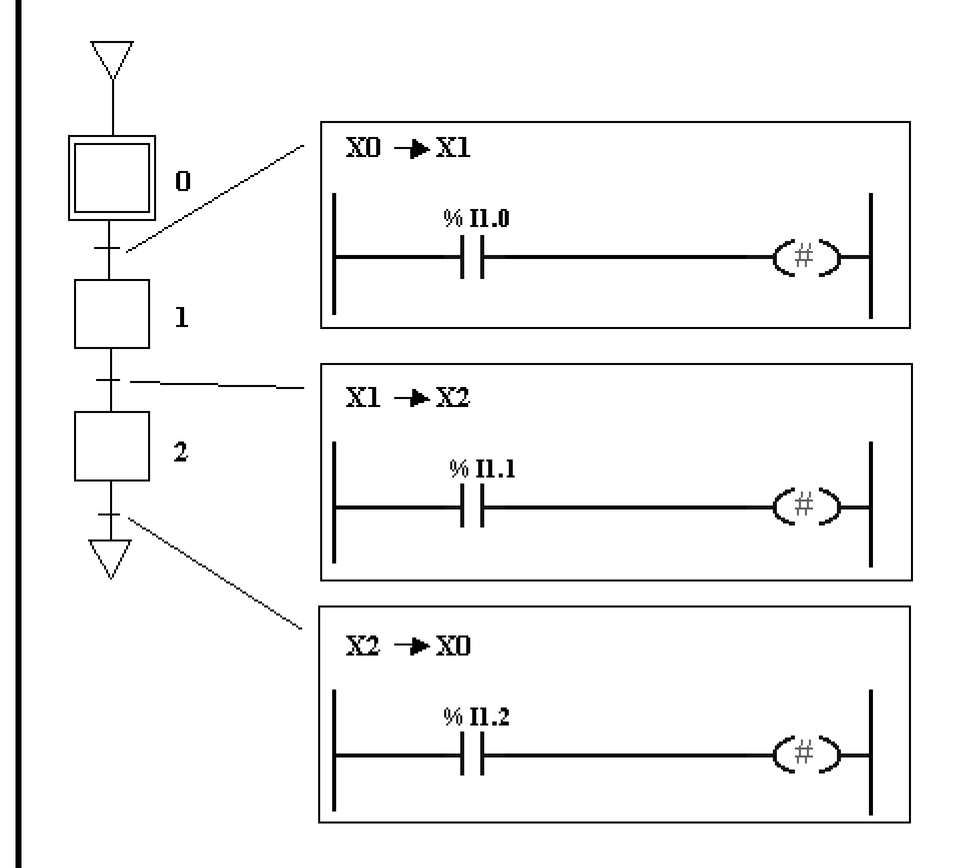
Ladder est une succession " de réseaux de contacts " véhiculant des informations logiques depuis les entrées vers les Le langage sorties. Les résultats dépendent des fonctions programmées.



Le Grafcet " Chart "



La construction du Grafcet se fait en ***CHART*** *comme ci-dessous :*



L'activation des sorties associées aux étapes du Grafcet s'effectue dans le ***POSTERIEUR***

Label 1

% X1 % Q2.0

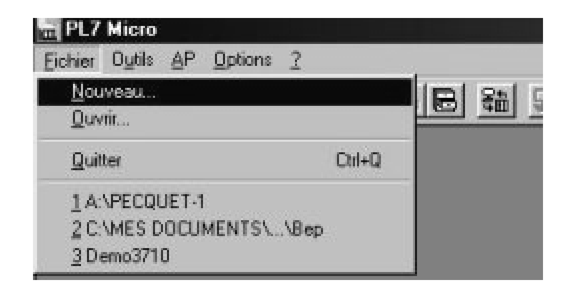
Label 2

% X2 % Q2.1

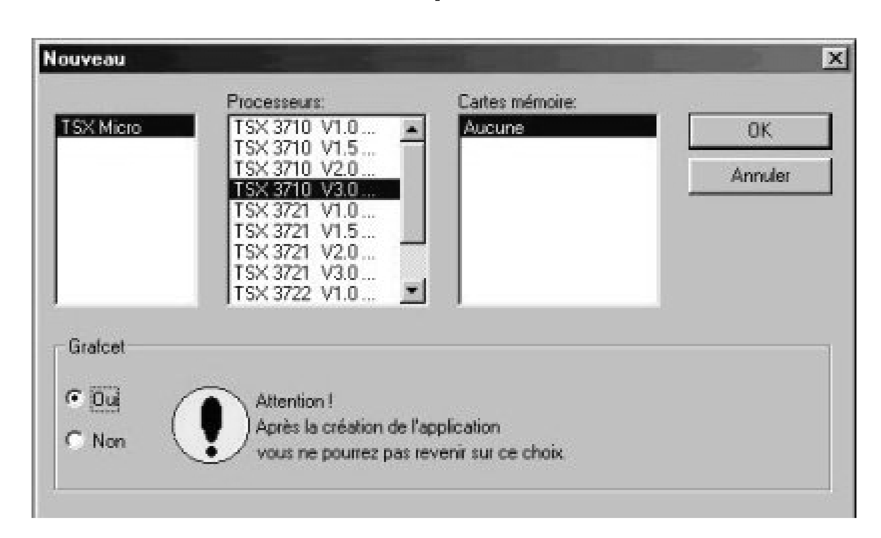
LES ETAPES DANS L ' UTILISATION DU LOGICIEL PL7 micro

Dans ce qui suite on présente en bref les étapes principales dans l'utilisation du logiciel PL7 micro, avec les écrans associés.

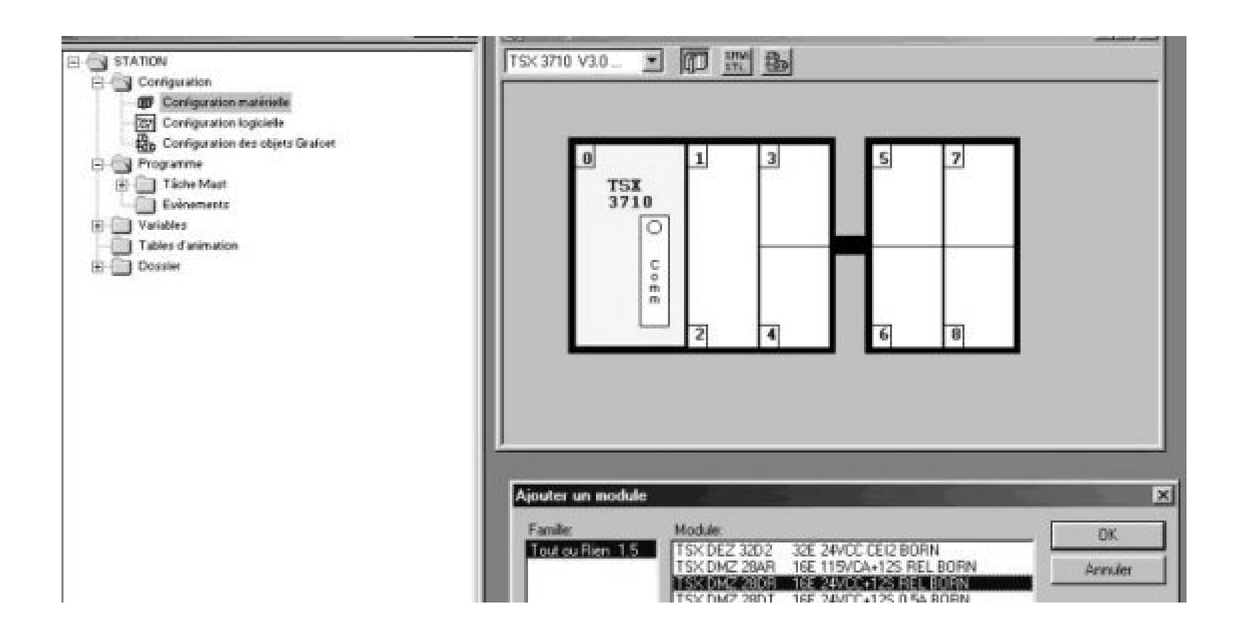
1)- Créer un nouveau projet

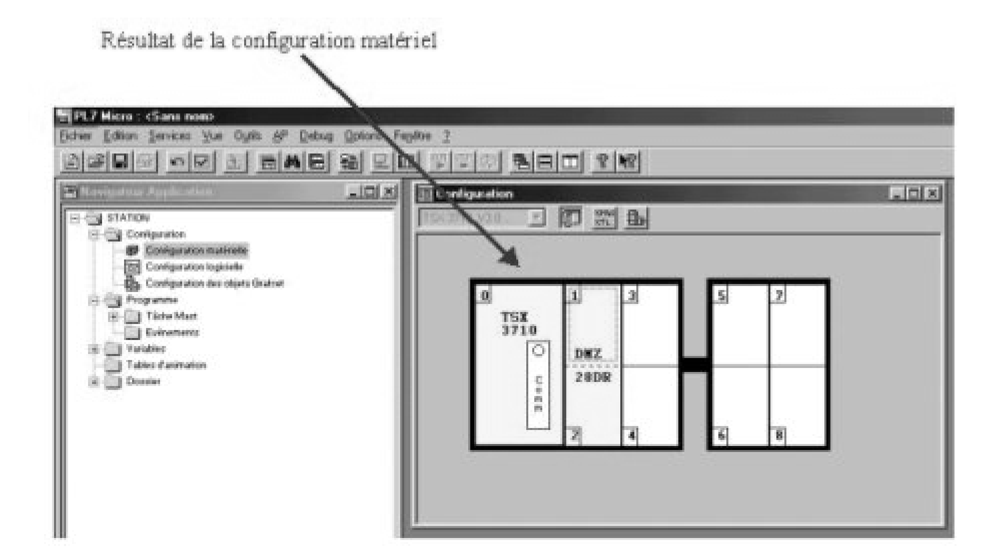


2)- Sélectionner la version d'automate que vous utilisé

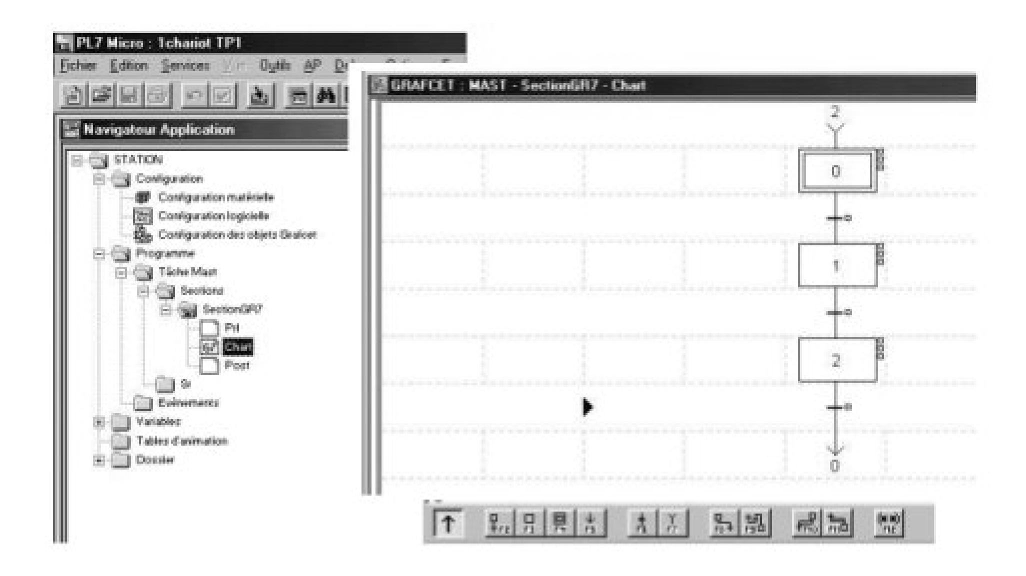


3)- Effectuer la configuration matérielle de l'automate à disposition

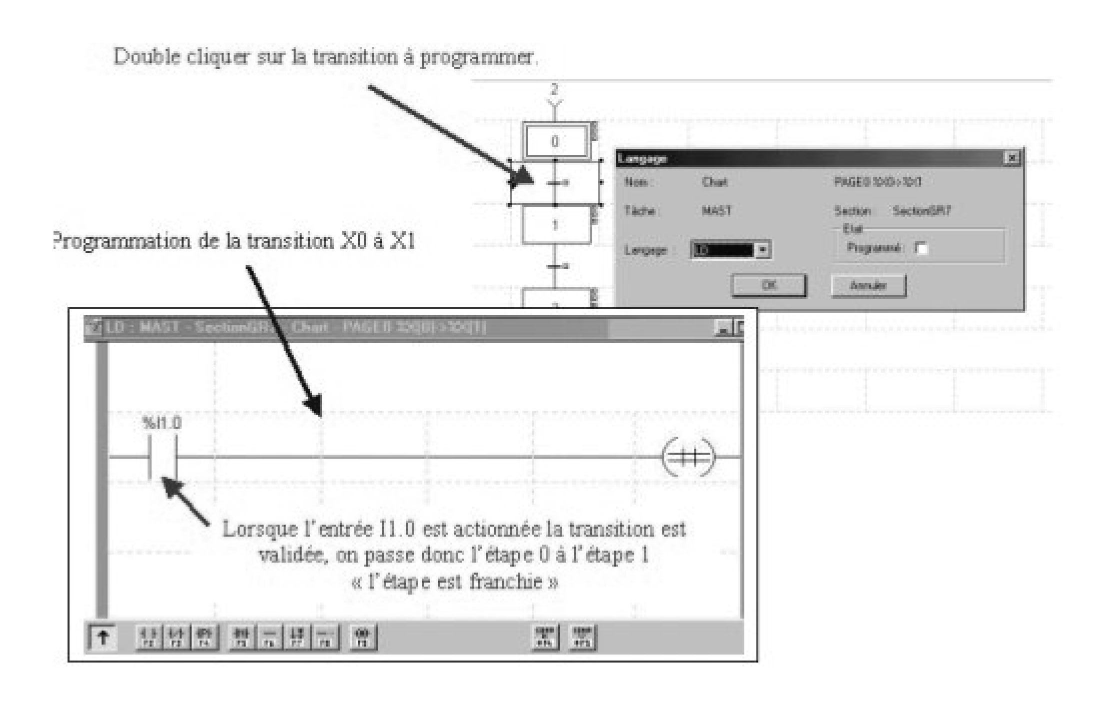




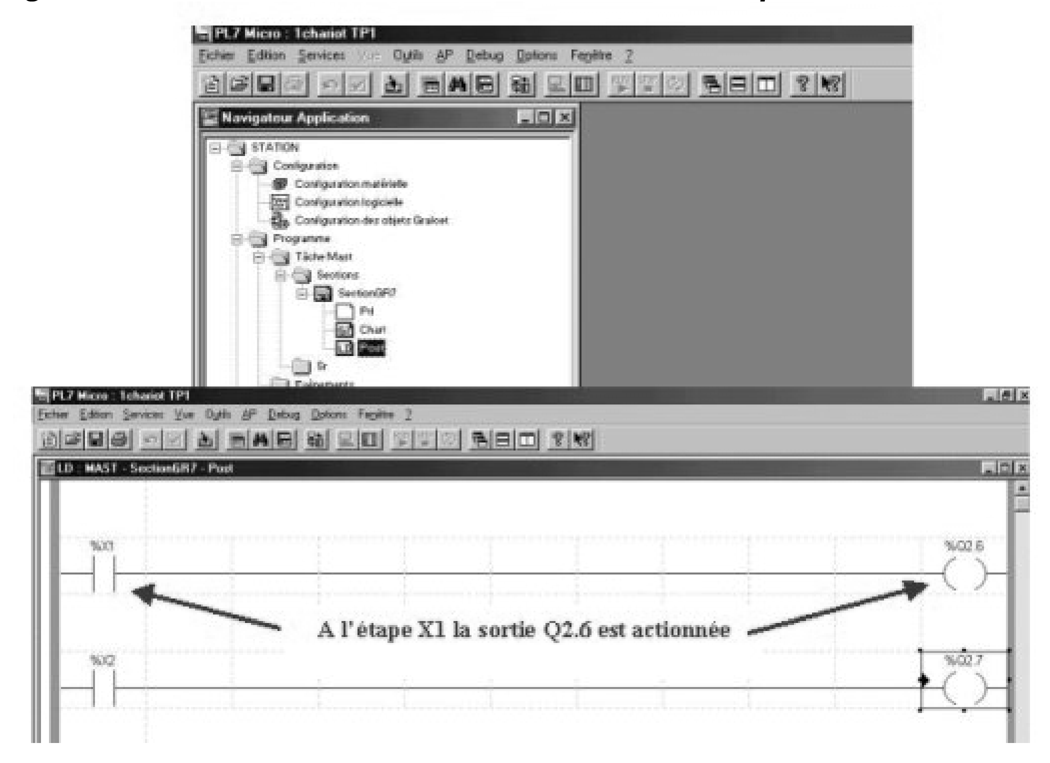
4)- Programmation du Chart " architecture du Grafcet "
a)- Réalisation de la forme du Grafcet



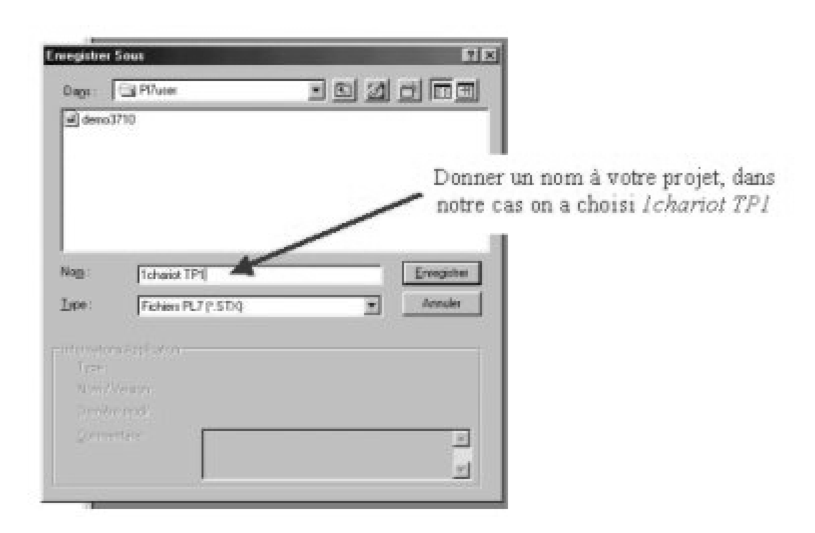
b)- Programmation des transitions



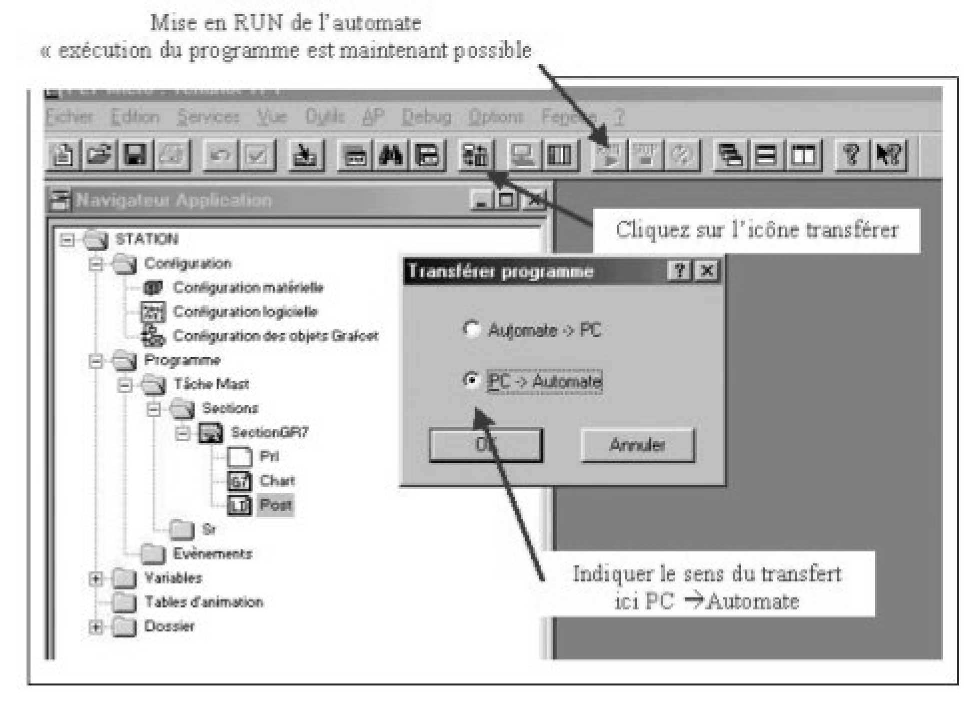
5)- Programmation du Post " actions associées aux étapes "



6)- Enregistrer votre projet



7)- Transférer votre programme du PC dans l'automate



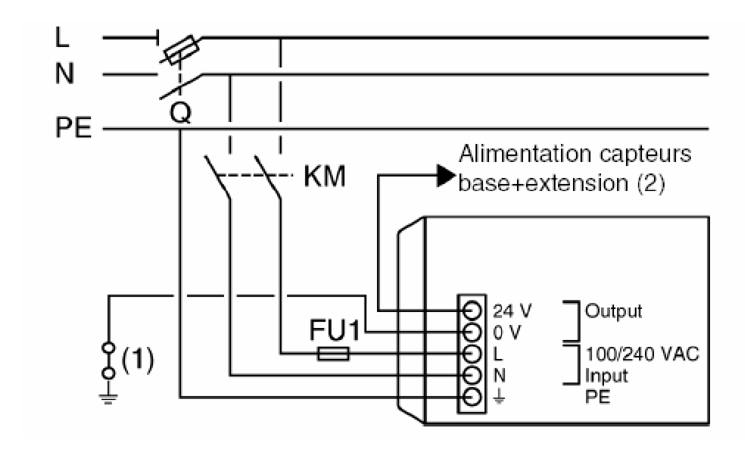
8)- Mettre l'automate en RUN et tester votre programme

10 RACCORDERDEMENT D'UN AUTOMATE PROGRAMMABLE

Pour raccorder un automate, il est recommandé de suivre :

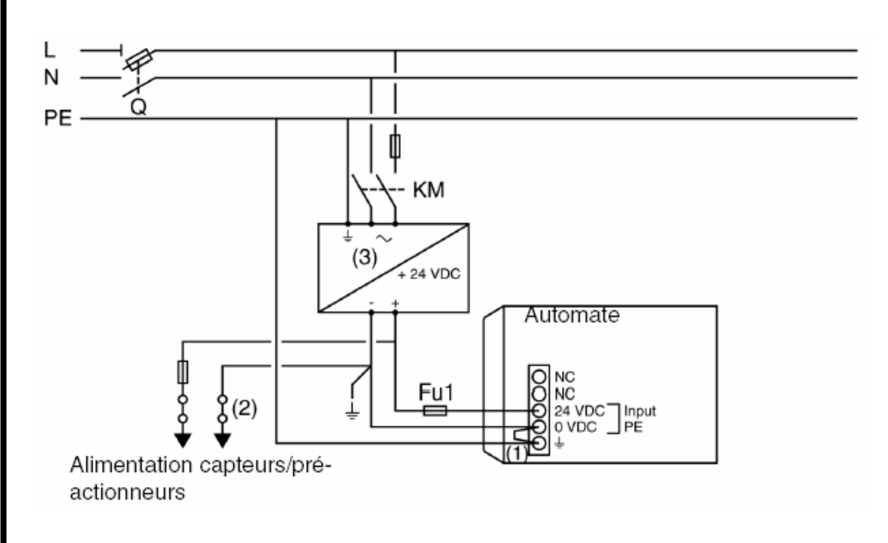
- *Les spécifications du fabricant*
- *La technique de raccordement*
- *De vérifier si les modules sont dans leurs embases respectives. Vérifier le type, le numéro du modèle et le diagramme de câblage. Vérifier l'emplacement des embases dans le document pour l'assignation des adresses d'E/S.*
- *De localiser le paquet de fils correspondant à chaque module et le diriger à travers le conduit à l'emplacement du module.*
  Identifier chacun des fils dans le paquet et s'assurer qu'ils correspondent à ce module en particulier.
- *En commençant avec le premier module, repérer le fil dans le paquée qui se branche à la borne la plus basse. Au point où le fil arrive à la même hauteur que le point de terminaison, plie le fil à angle droit vers la borne.*
- *De couper le fil pour qu'il dépasse de 6 mm du côté de la vis de la borne. Dégainer l'isolant du fil à approximativement 9 mm*
  Insérer le fil sous la plaque de la borne et serrer la vis.
- *Si deux modules ou plus utilisent la même source d'alimentation, on peut utiliser du cavalier «jumpers » pour le câblage de la source d'alimentation d'un module à l'autre.*
- *Si le câble blindé est utilisé, en brancher seulement un bout à la mise à la terre, préférablement au châssis. Ce branchement évitera toutes boucles possibles de retour de masse. L'autre bout doit être coupé et non branché.*
- *De répéter la procédure de câblage pour chaque fil du paquet jusqu'à ce que le câblage du module soit complété. Après que tous les fils aient été branchés, tirer doucement sur chacun pour s'assurer d'avoir un bon branchement.*
- *De répéter la procédure de câblage jusqu'à ce que tous les modules soient terminés.*

Raccordement d'un automate alimenté en alternatif 100-240 V



Q : sectionneur général,
KM : contacteur de ligne ou disjoncteur,
Fu1 : fusible 1 A temporisé.
(1) : barrette d'isolement pour recherche d'un défaut de mise à la masse,
(2) : ne pas dépasser 400 mA.

Raccordement des automates alimentés en continu



Q : sectionneur général,
KM : contacteur de ligne ou disjoncteur,
Fu1 : fusible 4A temporisé,
(1) : shunt externe fourni e monté sur l'AP. Ne doi pas être démonté,
(2) : barrette d'isolement pour recherche d'un défaut de mise à la masse. Il est nécessaire pour cela de supprimer le shunt externe, afin de déconnecter la borne d'alimentation de la masse automate.
(3) : utiliser une alimentation TSX SUP.

:

Module N° 17 :
AUTOMATE PROGRAMMABLE
GUIDE DES TRAVAUX PRATIQUES

I. TP 1 : schémas logiques de commande

I.1. Objectif(s) visé(s) :

- écrire la fonction logique associée a un schéma donne
- ayant le schéma de commande écrire la fonction logique associée

I.2. Durée du TP:

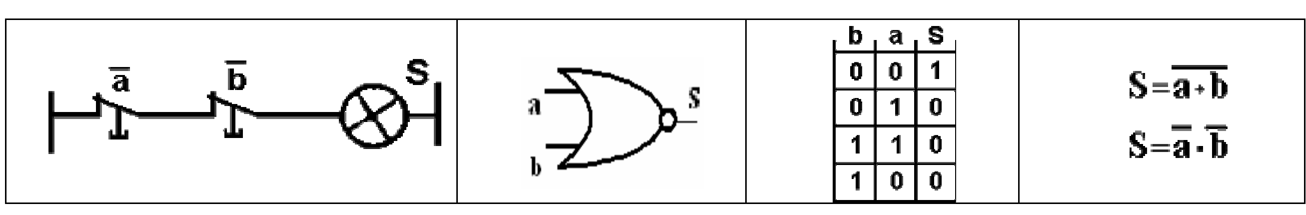
……………………………………………………………………………

I.3. Matériel (Équipement et matière d'œuvre) par équipe :

I.4. Description du TP :

On rappel :

Schéma	Symbole	Table de vérité	Équations
		a S 0 0 1 1	$S=a$
		a S 0 1 1 0	$S=\bar{a}$
		b a S 0 0 0 0 1 0 1 1 1 1 0 0	$S=a \cdot b$
		b a S 0 0 0 0 1 1 1 1 1 1 0 1	$S=a+b$
		b a S 0 0 0 0 1 1 1 1 0 1 0 0	$S=a \cdot \bar{b}$
		b a S 0 0 1 0 1 1 1 1 0 1 0 1	$S=\overline{a \cdot b}$ $S=\bar{a}+\bar{b}$

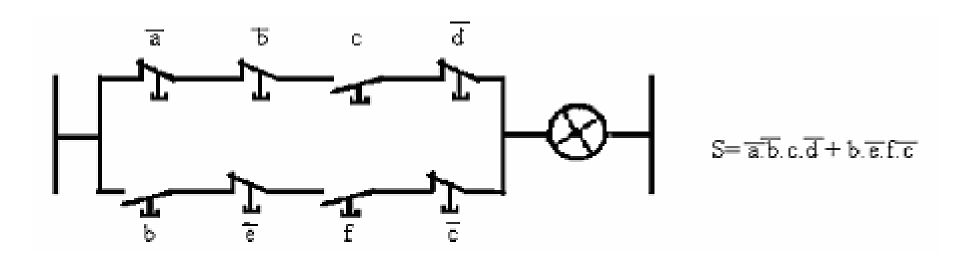


b	a	S
0	0	1
0	1	0
1	1	0
1	0	0

$S=\overline{a+b}$

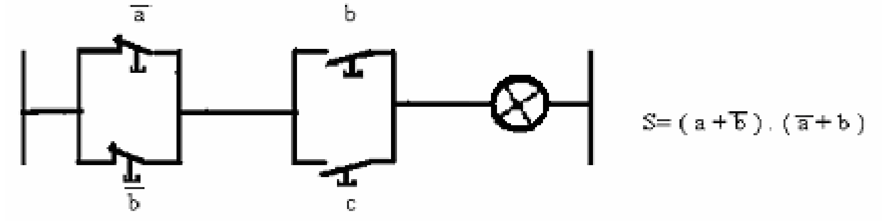
$S=\overline{a}\cdot\overline{b}$

Exemples



On dit : La lampe S s'allume si on ait une des deux conditions accomplie (fonction OU) :

- *contactes a inactive (normalement ferme), b inactive (normalement ferme), c active (normalement ouvert), d inactive (normalement ferme) – fonction ET*
- *contactes b active (normalement ouvert), e inactive (normalement ferme), f active (normalement ouvert), c inactive (normalement ferme) – fonction ET*



On dit : La lampe S s'allume si on ait les des deux conditions accomplie (fonction ET) :

- *soit le contacte a inactive (normalement ferme), soit le contact b inactive (normalement ferme) – fonction OU*
- *soit le contacte b active (normalement ouvert), soit le contact c active (normalement ouvert) – fonction OU*

I.5.Déroulementnt du TP

En utilisant les schéma de commande de base pour l'alimentation d'une charge électrique présentés dans l'introduction, et leurs fonctions logique associes, le stagiaire doit :

- *écrire la fonction logique associée a un schéma donne*
- *ayant le schéma de commande, écrire la fonction logique associée*

II. *TP 2 : architecture d'un API*

II.1. Objectif(s) visé(s) :

Mettre en évidence les différentes parties qui apparaisse dans l'architecture d'un API

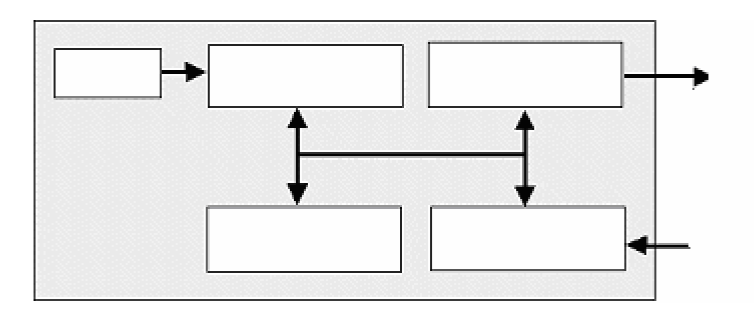
II.2. Durée du TP:

………………………………………………………………………………

II.3. Matériel (Équipement et matière d'œuvre) par équipe :

II.4. Description du TP :

II.5.Déroulement duTP



Dans le schéma antérieur positionner :

- *L'horloge*
- *Microprocesseur*
- *Interfaces d'entrée*
- *Interfaces de sortie*
- *Mémoires*
- *Commande de pré-actioneurs*
- *Dialogue homme machine/Etat du système*

Détailler le rôle et le fonctionnement du chacun de ces blocs

III. TP 3 : Interfaces d'entrées/sorties

III.1. Objectif(s) visé(s) :

- Reconnaître les différents types des interfaces TOR (tout ou rien)
- Savoir caractériser les interfaces

III.2. Durée du TP:

………………………………………………………………………………

III.3. Matériel (Équipement et matière d'œuvre) par équipe :

III.4. Description du TP :

III.5. Déroulement duTP

Du quel type sont chacun des interfaces suivantes. Expliquer en bref leur fonctionnement

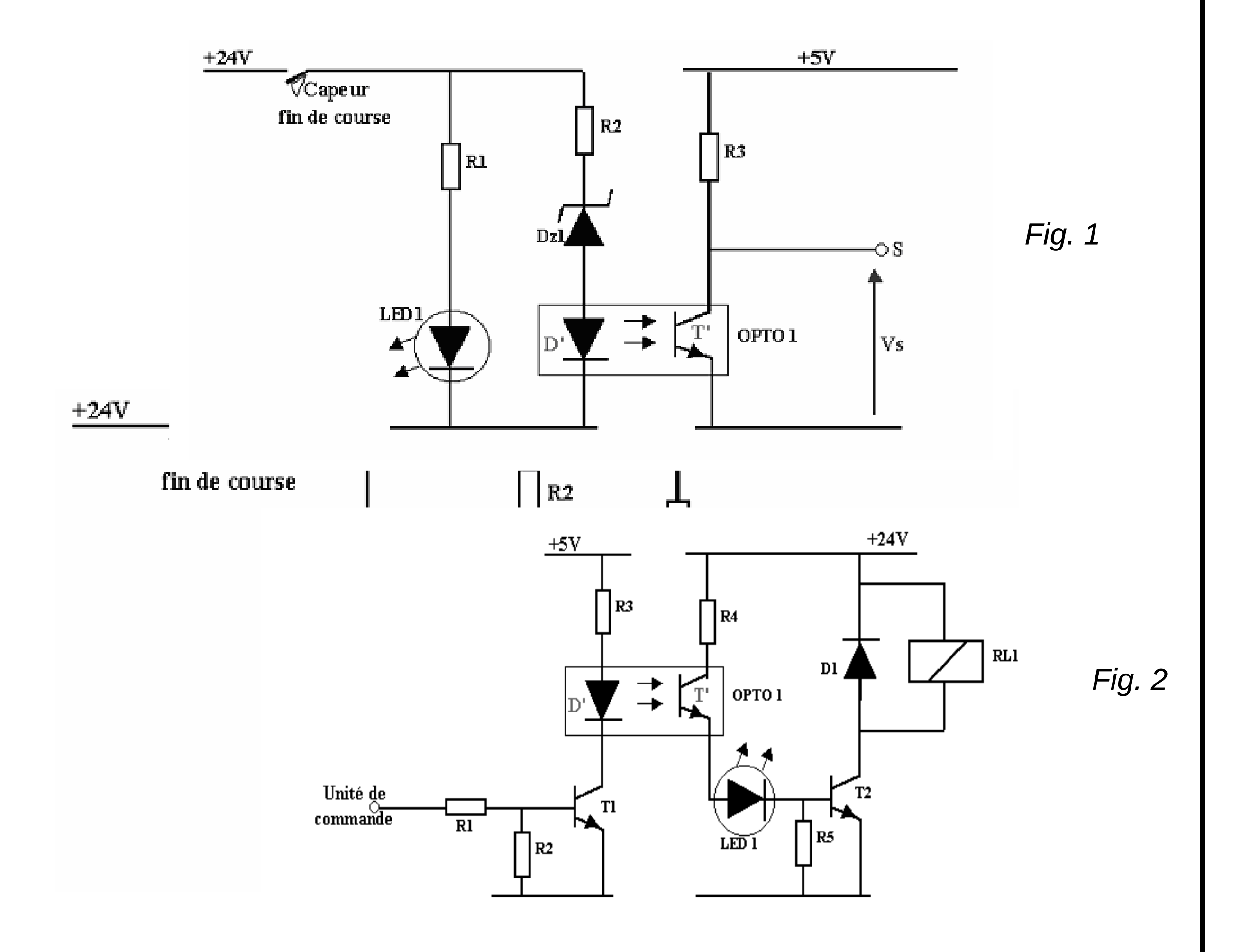


Fig. 1

Fig. 2

IV. *TP 4 : Fonctionnement d'un outillage semi-automatisé*

IV.1. Objectif(s) visé(s) :
Etre capable de décrire la séquence d'automatisation pour un exemple simple

IV.2. Durée du TP:

……………………………………………………………………………

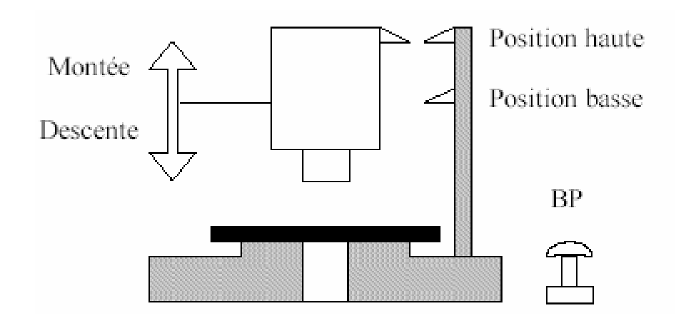
IV.3. Matériel (Équipement et matière d'œuvre) par équipe :

IV.4. Description du TP :
IV.5. Déroulement du TP

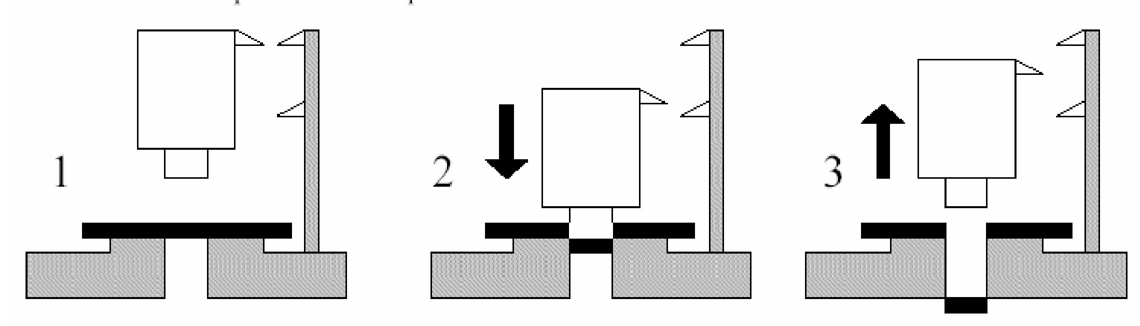
On lie attentivement les explications pour les quatre applications données comme exemple dans le chapitre 3.

On regarde les deux figures suivantes représentant une poinçonneuse semi-automatique et ses trois états dans un cycles de fonctionnement :

- *la poinçonneuse est au repos ou encore en position haute*
- *le poinçon descend*
- *le poinçon monte*



- Cette machine possède 3 comportements différents :



Essayer expliquer le fonctionnement, en corrélant les informations provenant aux trois entrées (capteur position haute, capteur position basse, bouton poussoir de START (BP), avec les deux commandes de sortie : déplacement en bas, déplacement en haut)

V. TP V : : Fonctionnement d'un outillage automatisé

V.1. Objectif(s) visé(s) :

Etre capable de décrire la séquence d'automatisation pour un exemple plus élaboré

V.2. Durée du TP:

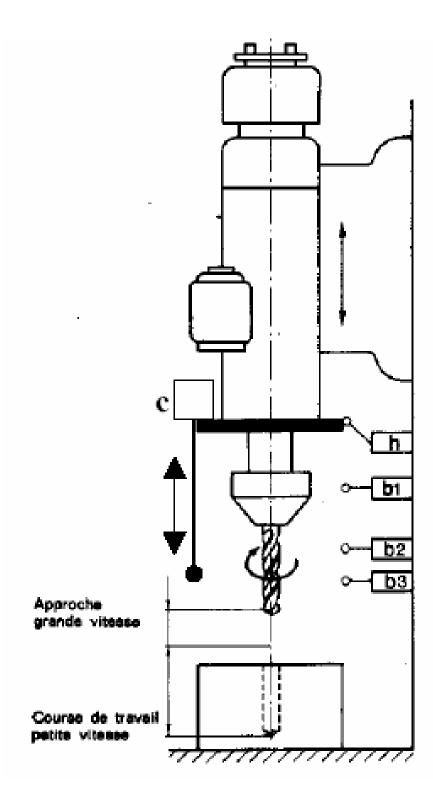
………………………………………………………………………………

V.3. Matériel (Équipement et matière d'œuvre) par équipe :

V.4. Description du TP :

V.5.Déroulement duTP

On refait les étapes du rationnement pour le TP antérieur, mais pour une perceuse avec ou sans débourrage



Cycle sans débourrage :

- *descente en grande vitesse jusque b1*
- *descente en petite vitesse jusque b3*
- *remontée en grande vitesse jusqu'à h*

Cycle avec débourrage :

- *descente en grande vitesse jusque b1,*
- *cycle active lorsque le capteur c entre en contact avant l'enclenchement du contact b2,*
- *remontée en grande vitesse de la broche a une position intermédiaire b1,*
- *descente en petite vitesse jusque b3*
- *remontée en grande vitesse jusqu'a h*

VI. TP 6 : Ecrire des programmes en langage à contacts (ladder) simples

VI.1. Objectif(s) visé(s) :

Maîtriser les connaissances de base d'après le ladder

VI.2. Durée du TP:

……………………………………………………………………………

VI.3. Matériel (Équipement et matière d'œuvre) par équipe :

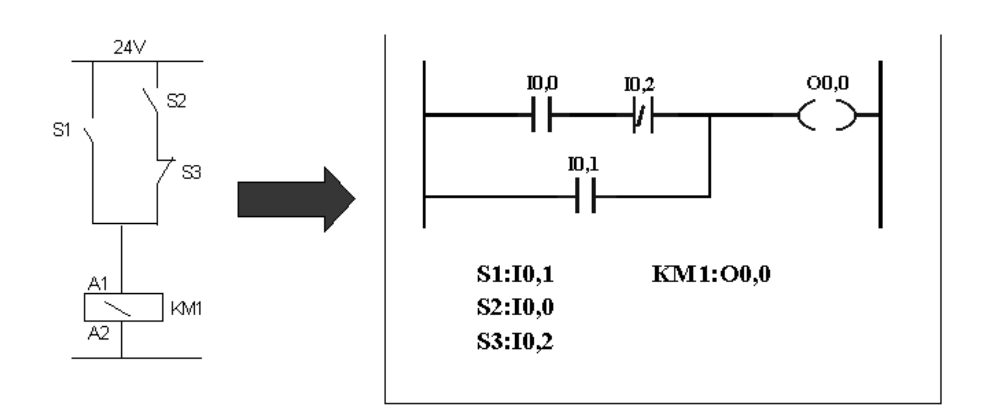
VI.4. Description du TP :

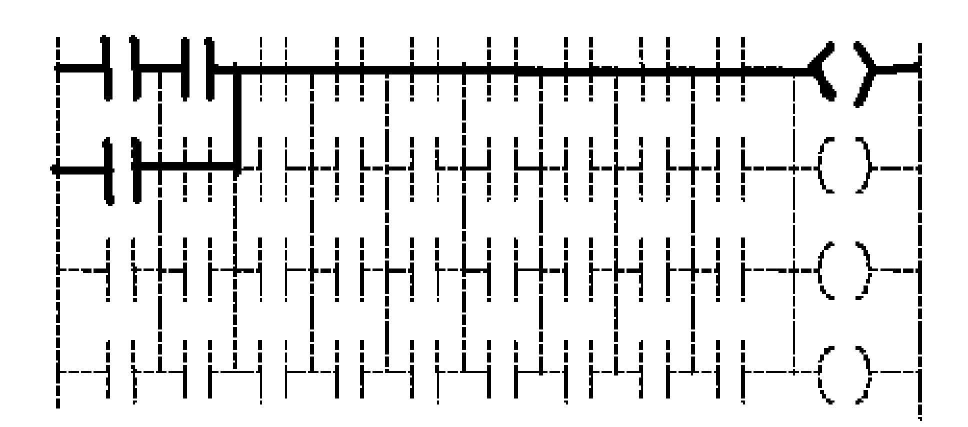
VI.5. Déroulement duTP

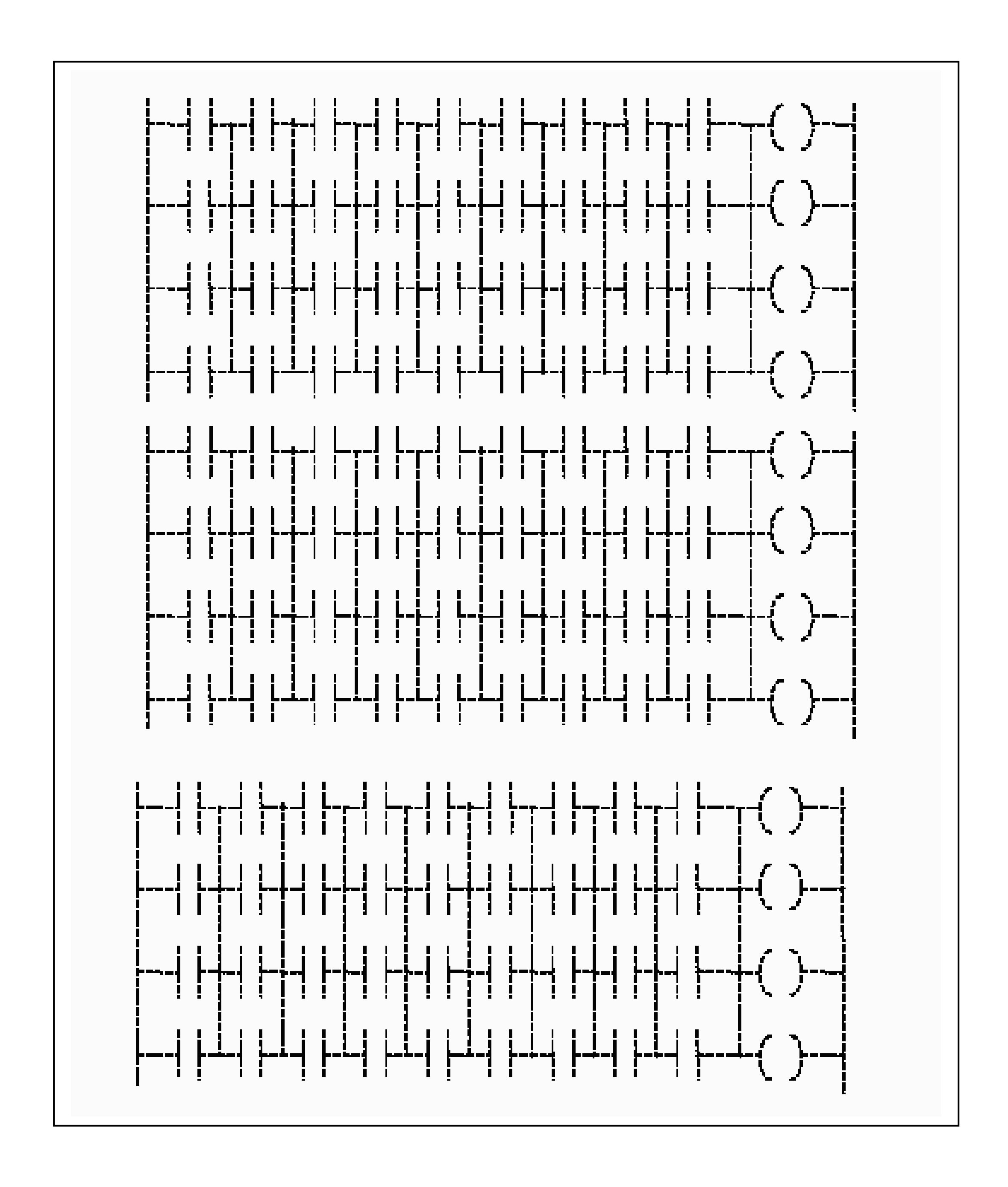
Pour les schéma base de commande du TP1 écrire les programmes ladder correspondantes en remplissant le formulaire de la fig.1 (voire page suivante)

Exemple :

Fig.1







VII. *TP 7 : Etablir le grafcet d'un automatisme simpe*

VII.1. Objectif(s) visé(s) :
Maîtriser les connaissances de base d'après le grafcet

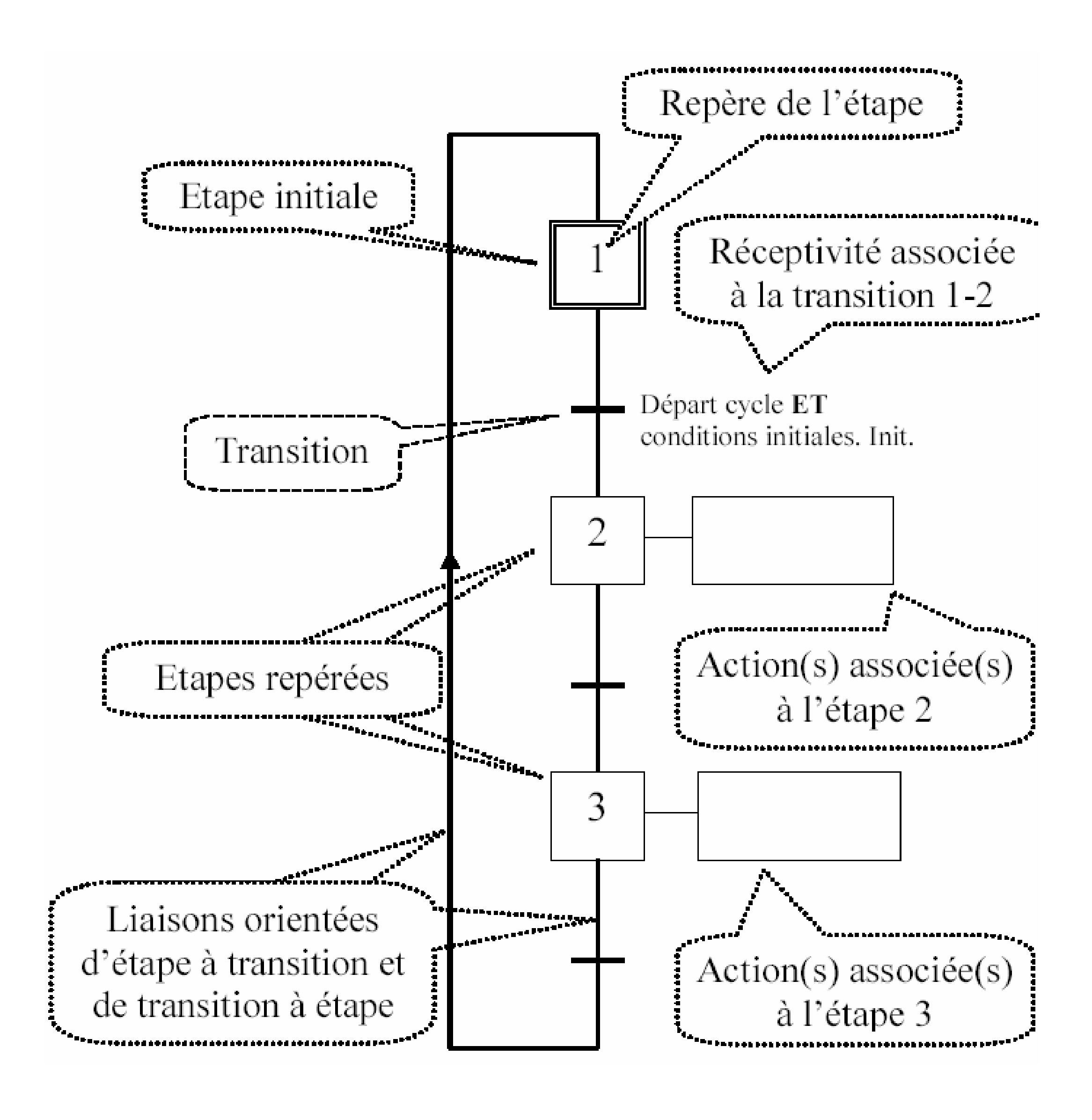
VII.2. Durée du TP:

…………………………………………………………………………

VII.3. Matériel (Équipement et matière d'œuvre) par équipe :

VII.4. Description du TP :
VII.5. Déroulement duTP

Rappel les éléments d'un grafcet



En utilisant l'exemple antérieur tracez le grafcet pour la poinçonneuse duTP4

VIII. *TP 8 : Etablir le grafcet d'un exemple plus élabore*

VIII.1. Objectif(s) visé(s) :

Maîtriser la technique des grafcets pour des exemples plus élaborés

VIII.2. Durée du TP:

……………………………………………………………………………

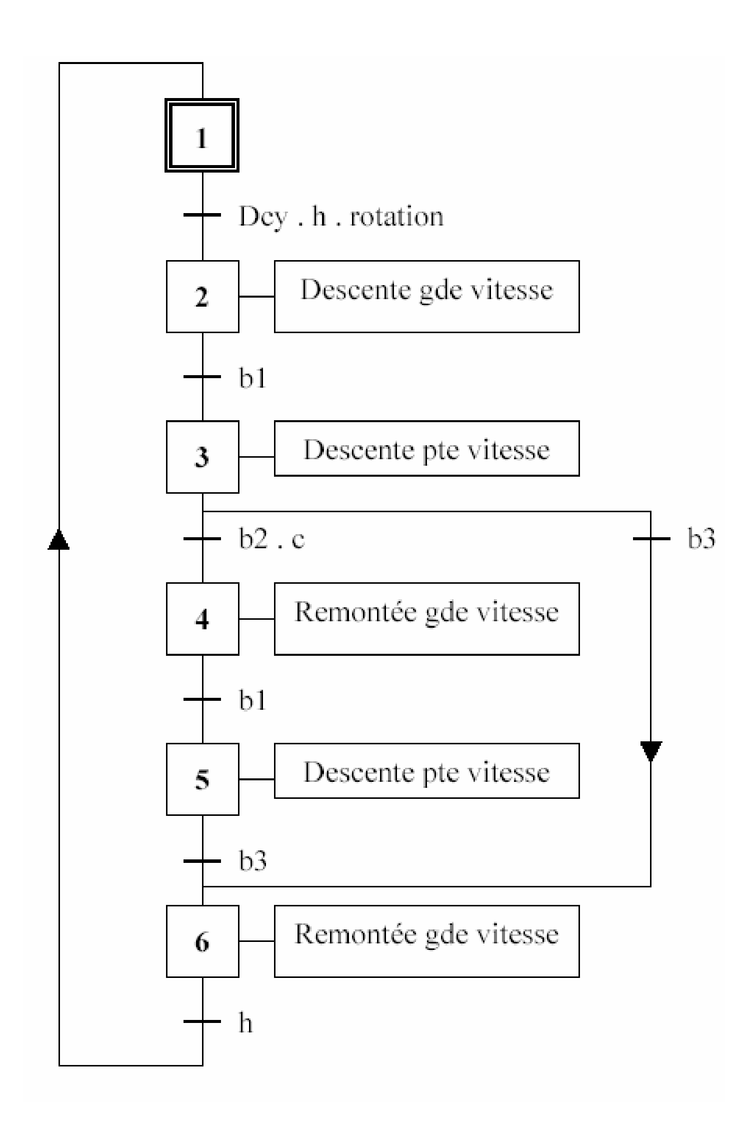
VIII.3. Matériel (Équipement et matière d'œuvre) par équipe :

VIII.4. Description du TP :

VIII.5.Déroulement du TP

Traces le grafcet de la perceuse du TP5, en utilisant les indications du TP7.

Solution :



IX. *TP 9 : Utilisation de la console opérateur*

IX.1. Objectif(s) visé(s) :
Savoir utiliser la console opérateur

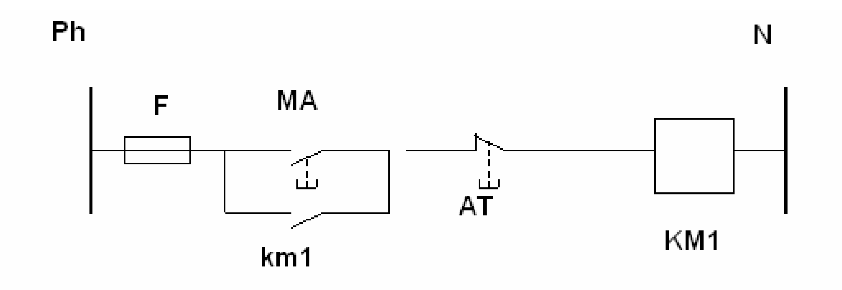
IX.2. Durée du TP:

…………………………………………………………………………….

IX.3. Matériel (Équipement et matière d'œuvre) par équipe :

IX.4. Description du TP :
IX.5. Déroulement duTP

Etablir le programme a contacts (ladder) pour l'exemple suivantes (démarrage direct) :



En utilisant la console de programmation de l'automate essayer d'introduire le programme dans l'automate.

X. TP 10 : Utilisation du logiciel PL7-micro

X.1. Objectif(s) visé(s) :
Savoir utiliser le logiciel PL-7 micro

X.2. Durée du TP:

…………………………………………………………………………….

X.3. Matériel (Équipement et matière d'œuvre) par équipe :

X.4. Description du TP :
X.5. Déroulement du TP

Reprenez l'exemple du TP9, en essayant faire la programmation de l'automate en utilisant ce fois ci le logiciel PL7-micro, en suivant les étapes décrites dans le chapitre 9.

XI. TP 11 : Utilisation de l'automate dans un régulation pressostatique

XI.1. Objectif(s) visé(s) :

- Application directe de l'automate dans le froid

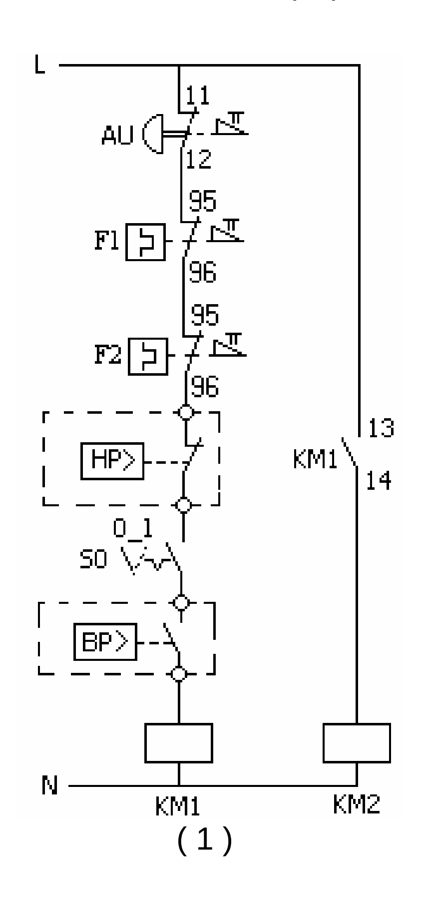
XI.2. Durée du TP:

……………………………………………………………………………

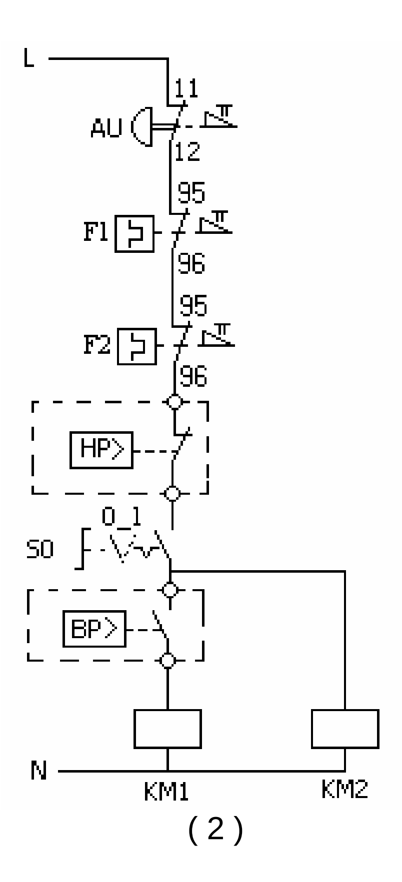
XI.3. Matériel (Équipement et matière d'œuvre) par équipe :

XI.4. Description du TP :

En prenant l'exemple d'une régulation basée sur l'utilisation d'un pressostat BP de régulation avec les deux variantes : ventilateur fonctionnant dans le même temps avec l'agrégat de condensation (1), et ventilation continue (2)



(1)



(2)

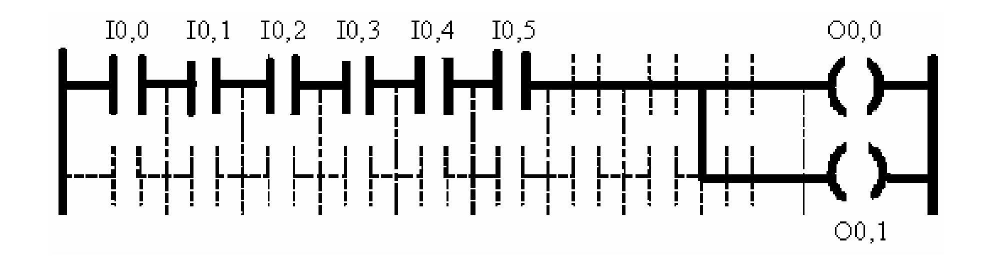
Légende:

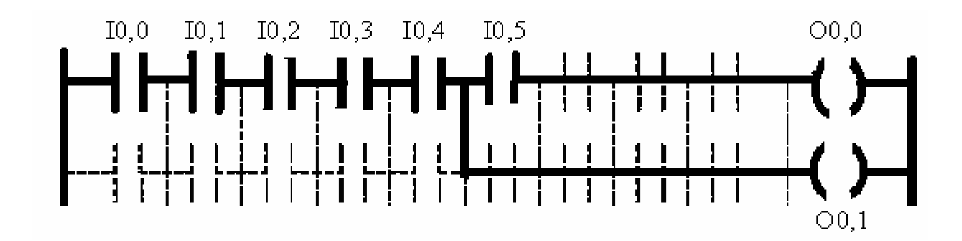
KM1 = Contacteur du groupe de condensation
KM2 = Contacteur du moteur de l'évaporateur
AU = Arrêt d'urgence
F1 = Relais thermique du groupe de condensation
F2 = Relais thermique du moteur de l'évaporateur
HP> = Pressostat HP de sécurité
S0 = Commutateur marche/arrêt
T> = Thermostat de régulation
BP> = Pressostat BP de régulation

XI.5.Déroulement du TP

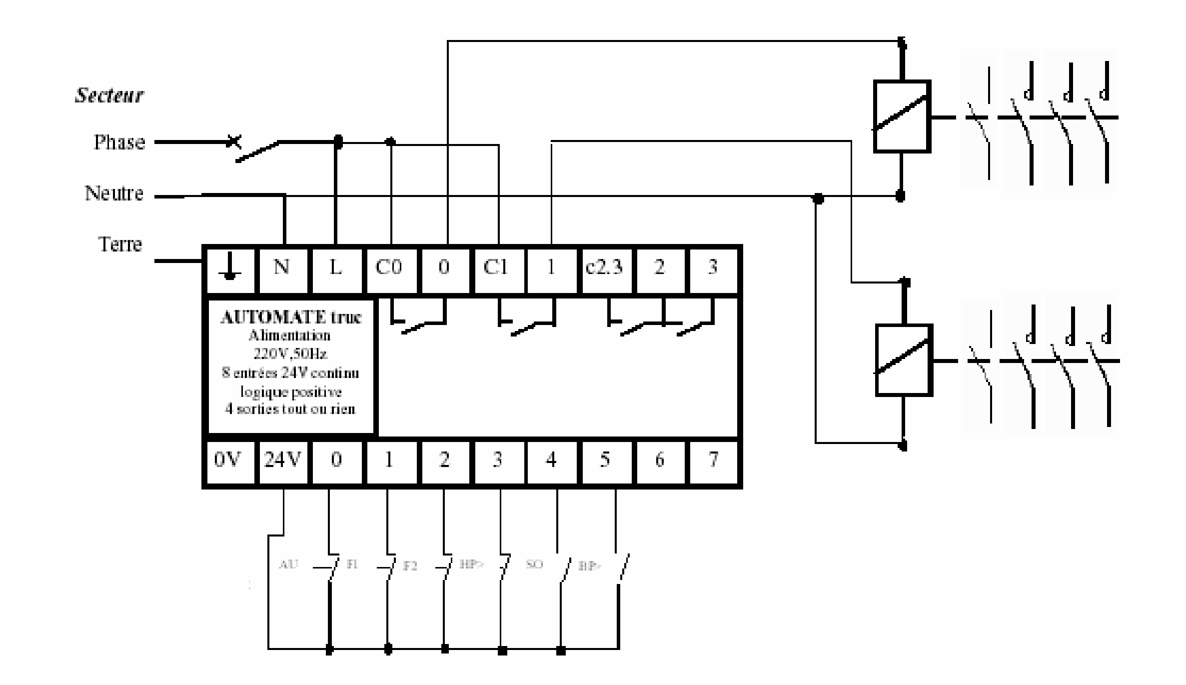
En utilisant les formulaires de programmation du TP4, on dessine les diagrammes ladder pour les deux types de régulation.

Solution





Les schémas de connexions des entrées et des sorties :



Liste des références bibliographiques.

Ouvrage	***Auteur***	***Edition***
/		

NB : Outre les ouvrages, la liste peut comporter toutes autres ressources jugées utiles (Sites Internet, Catalogues constructeurs, Cassettes, CD,…)